COURSE OUTLINE

MS-GH500T00: GITHUB ADVANCED SECURITY

DURATION	LEVEL	TECHNOLOGY	DELIVERY METHOD	TRAINING CREDITS
1 Day	Intermediate	Microsoft GitHub	Instructor-led	NA

INTRODUCTION

GitHub Advanced Security (GHAS) plays a crucial role in enhancing the security posture of software development projects on GitHub. It provides a comprehensive set of tools and features designed to identify and address security vulnerabilities throughout the development lifecycle. By integrating security directly into the development process with GHAS, your team can build more secure and reliable software. The course will explore how to utilize GHAS to maximize security impact and understand GHAS and its role in the security ecosystem.

AUDIENCE PROFILE

This course in intended for students who want to understand and implement advanced security practices with the help of GitHub Advanced Security (GHAS). They will learn how to significantly enhance software development processes and create a more resilient and secure development ecosystem using developer-first solutions to unlock the ability to keep code, supply chain, and secrets secure before you push to production. They will learn how GHAS gives security teams visibility into the cross-organizational security posture and supply chain and unparalleled access to curated security intelligence from millions of developers and security researchers around the world.

PREREQUISITES

Before attending this course, delegates must have:

- A GitHub account — essential for hands-on labs and exercises.
- Basic understanding of GitHub fundamentals, such as repositories, commits, and pull requests.
- Familiarity with software development workflows, especially around version control and collaboration.

COURSE OBJECTIVES

After attending this course, delegates will be able to:

- Understand and configure GitHub Advanced Security (GHAS) features across repositories.
- Implement Dependabot to automate dependency updates and manage vulnerabilities.
- Set up and manage secret scanning to detect and protect sensitive information in your codebase.
- Configure and use code scanning with CodeQL to identify and remediate security issues.
- Analyze and interpret scan results to make informed security decisions.
- Administer security policies and manage sensitive data within GitHub effectively.

COURSE CONTENT

Module 1: Introduction to GitHub Advanced Security

This module will help you become familiar with GitHub's Advanced Security features (GHAS) and best practices. As you learn about these features, you'll identify critical areas for eliminating security gaps.

Lessons

- Introduction
- Define GHAS and the importance of its integral features
- How to utilize GHAS to get the most impact
- Understand GHAS and its role in the security ecosystem
- Module assessment
- Summary

After completing this module, students will be able to:

- Define GHAS and the importance of the integral features such as Secret scanning, Code scanning, and Dependabot
- Know how to utilize GHAS to maximize security impact
- Understand GHAS and its role in the security ecosystem

Module 2: Configure Dependabot security updates on your GitHub repo

Manage your dependencies with GitHub Dependabot.

Lessons

- Introduction
- Manage your dependencies on GitHub
- Dependabot alerts
- Dependabot security updates
- Manage Dependabot notifications and reports
- Dependency review
- Exercise - Configure Dependabot security updates
- Module assessment
- Summary

After completing this module, students will be able to:

- Describe the available tools for managing vulnerable dependencies on GitHub.
- Enable and configure Dependabot alerts.
- Identify the permissions and roles required to view and enable Dependabot alerts.
- Enable and configure Dependabot security updates.
- Identify, review, and address vulnerable dependencies.
- Explain how to use GraphQL API to retrieve vulnerability information.
- Explain how to configure notifications for vulnerable dependencies.

Module 3: Configure and use secret scanning in your GitHub repository

Understand how secret scanning works to configure and use it efficiently.

Lessons

- Introduction
- What is secret scanning?
- Configure secret scanning
- Use secret scanning
- Exercise
- Module assessment
- Summary

After completing this module, students will be able to:

- Describe secret scanning.
- Configure secret scanning.
- Use secret scanning.

Module 4: Configure code scanning on GitHub

This module introduces you to code scanning and its features. You'll learn how to implement code scanning using CodeQL, third party tools, and GitHub Actions..

Lessons

- Introduction
- What is code scanning?
- Enable code scanning with third party tools
- Configure code scanning
- Configure code scanning exercise
- Module assessment
- Summary

After completing this module, students will be able to:

- Describe code scanning.
- List the steps for enabling code scanning in a repository.
- List the steps for enabling code scanning with third-party analysis.
- Contrast how to implement CodeQL analysis in a GitHub Actions workflow versus a third-party continuous integration (CI) tool.
- Explain how to configure code scanning on a repository using triggering events.
- Contrast the frequency of code scanning workflows (scheduled vs triggered by events).

Module 5: Identify security vulnerabilities in your codebase by using CodeQL

In this module, you learn about CodeQL and how you can use it to analyze the code in your GitHub repository and identify security vulnerabilities.

Lessons

- Introduction
- Prepare a database for CodeQL
- Run CodeQL in a database
- Understand CodeQL results
- Troubleshoot CodeQL results
- Module assessment
- Summary

After completing this module, students will be able to:

- Create a database by using CodeQL to extract a single relational representation of each source file in the codebase.
- Run CodeQL in a database to find problems in your source code and find potential security vulnerabilities.
- Understand CodeQL scan results by using GitHub-created queries or your own custom queries.

Module 6: Management and customization considerations with GitHub Copilot

In this module, we explore management and customization considerations with GitHub Copilot.

Lessons

- Introduction
- Explore GitHub Copilot plans and their associated management and customization features
- Explore contractual protections in GitHub Copilot and disabling matching public code
- Manage content exclusions
- Troubleshoot common problems with GitHub Copilot
- Module assessment
- Summary

By the end of this module, you will:

- Understand the GitHub Copilot plans and their associated management and customization features.
- Gain insight into the contractual protections in GitHub Copilot and disabling matching public code.
- Know how to manage content exclusions.
- Recognize common problems with GitHub Copilot and their solutions.

Module 7: Developer use cases for AI with GitHub Copilot

This module explores how GitHub Copilot streamlines developer productivity through AI-powered features. It enhances the Software Development Life Cycle (SDLC), aligns with developer preferences, and identifies key limitations. Additionally, it measures productivity gains effectively.

Lessons

- Introduction
- Boost developer productivity with AI
- Align with developer preferences
- AI in the Software Development Lifecycle (SDLC)
- Understand limitations and measure impact
- Module assessment
- Summary

By the end of this module, you're able to:

- Identify specific ways GitHub Copilot integrates seamlessly into developer workflows, enhancing the overall development experience and supporting individual coding preferences.
- Explore GitHub Copilot's potential impact on different stages of the Software Development Lifecycle.
- Evaluate the limitations of AI-assisted coding and measure its impact on development efficiency.

Module 8: Develop unit tests using GitHub Copilot tools

This module explores using GitHub Copilot and GitHub Copilot Chat to create unit tests. Exercises provide practical experience creating unit test projects and running unit tests in Visual Studio Code.

Lessons

- Introduction
- Examine the unit testing tools and environment
- Create unit tests using the Generate Tests smart action
- Create unit tests using Inline Chat
- Create unit tests using Chat view modes
- Exercise - Develop unit tests using GitHub Copilot
- Module assessment
- Summary

In this module, you'll learn how to:

- Create unit tests using the GitHub Copilot and GitHub Copilot Chat extensions for Visual Studio Code.
- Create unit tests that target edge cases and specific conditions using the GitHub Copilot and GitHub Copilot Chat extensions for Visual Studio Code.
- Use Visual Studio Code, the .NET SDK, and the C# Dev Kit extension to create a test project and verify that your unit tests build and run successfully.

ASSOCIATED CERTIFICATIONS & EXAM

This course will prepare delegates to write the GitHub Advanced Security exam.