

AZ-400.2

Module 01:

Implementing Continuous Integration in an Azure DevOps Pipeline



Lesson 01: Continuous Integration Overview

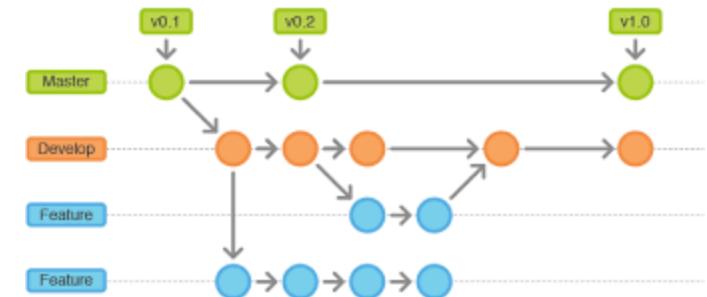


Lesson 1 Overview

- Introduction to Continuous Integration
- The Four Pillars of Continuous Integration
- Benefits of Continuous Integration
- Continuous Integration Implementation Challenges
- Implementing Continuous Integration in Azure DevOps
- Using Variables to Avoid Hard-coded Values
- Build Number Formatting and Build Status
- Build Authorizations, Timeouts, and Badges
- Configuring Build Retention
- Lab - Enabling Continuous Integration with Azure Pipelines

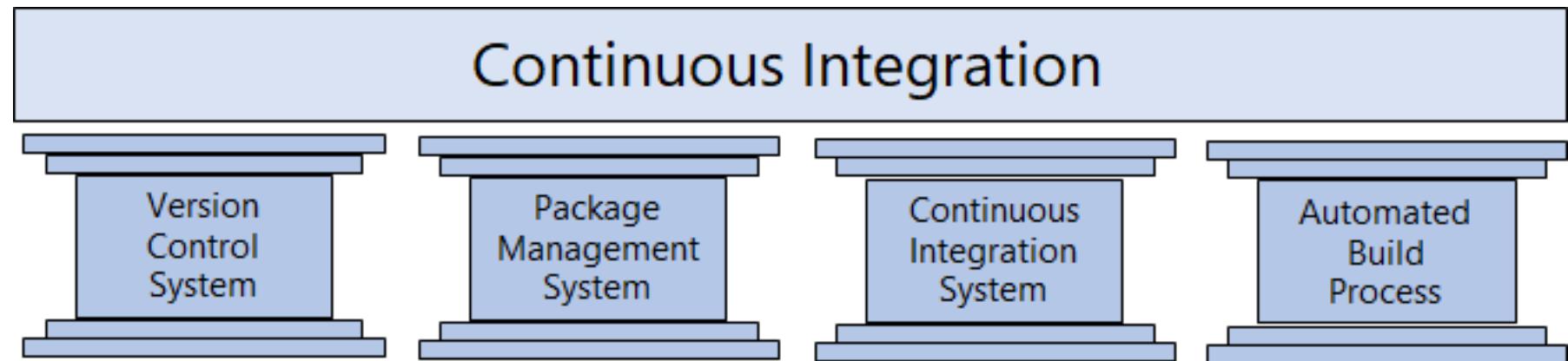
Introduction to Continuous Integration

- Continuous Integration (CI) is the process of automating the build and testing of code.
- CI encourages developers to share their code and unit tests by merging their changes into the shared version control repository.
- When a change is detected it triggers an automated build system. The code is built using a build definition. Developers respond to any issues or bugs.
- CI keeps the master branch clean ensuring bugs are caught earlier in the development cycle, which makes them less expensive to fix.



The Four Pillars of Continuous Integration

- A Version Control System manages changes to your source code over time.
- A Package Management System is used to install, uninstall and manage software packages.
- A Continuous Integration System merges all developer working copies to a shared mainline several times a day.
- An Automated Build Process creates a software build including compiling, packaging, and running automated tests.



Benefits of Continuous Integration

- Improving code quality based on rapid feedback
- Triggering automated testing for every code change
- Reducing build times for rapid feedback and early detection of problems (risk reduction)
- Better management of technical debt and code analysis
- Reducing long, difficult, and bug-inducing merges
- Increasing confidence in codebase health long before production deployment
- Rapid feedback to the developer

Demonstration: Implementing Continuous Integration in Azure DevOps

... > FoodApp-ASP.NET Core-T02-Demo01-import

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

Pipeline Build pipeline

Get sources FoodApp master

Agent job 1 Run on agent

Restore .NET Core

Build .NET Core

Test Disabled: .NET Core

Publish .NET Core

Publish Artifact Publish build artifacts

Select a source

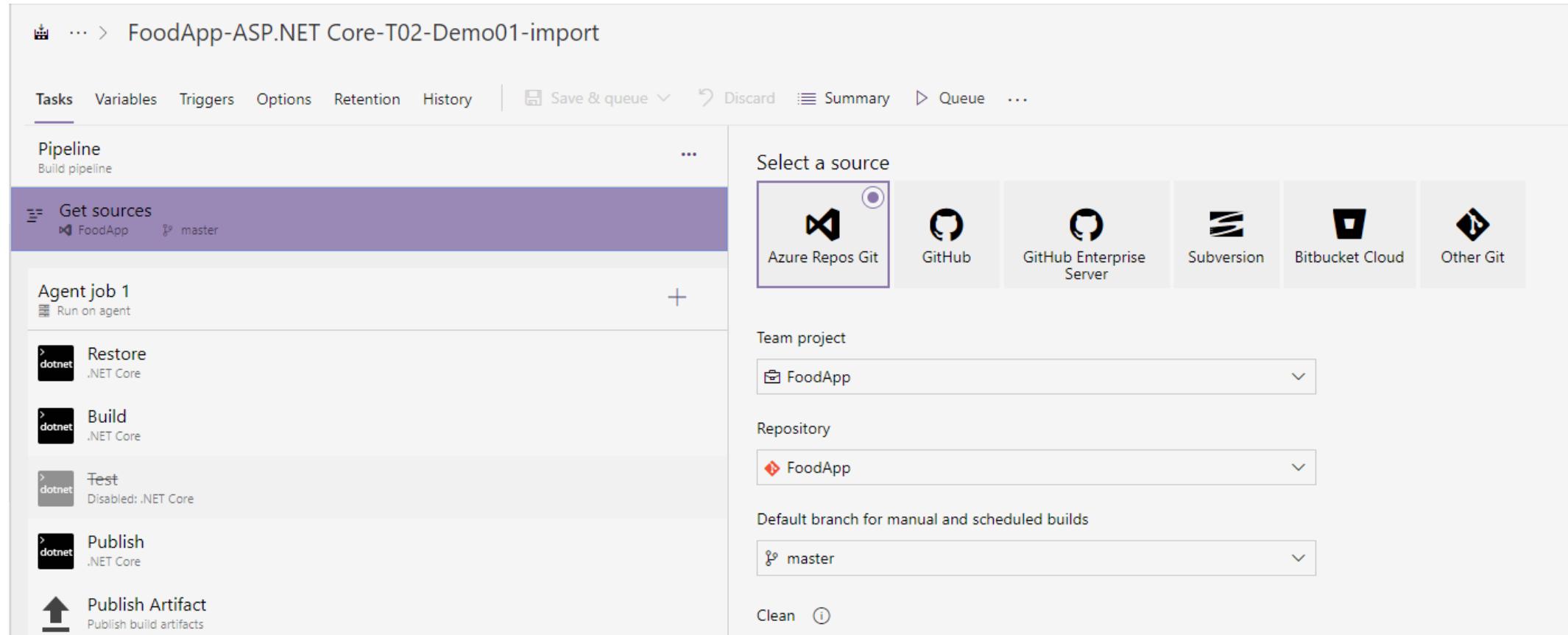
Azure Repos Git GitHub GitHub Enterprise Server Subversion Bitbucket Cloud Other Git

Team project FoodApp

Repository FoodApp

Default branch for manual and scheduled builds master

Clean



Authorization and Timeouts

Authorization and Timeouts (scope, job timeout, cancel job timeout)

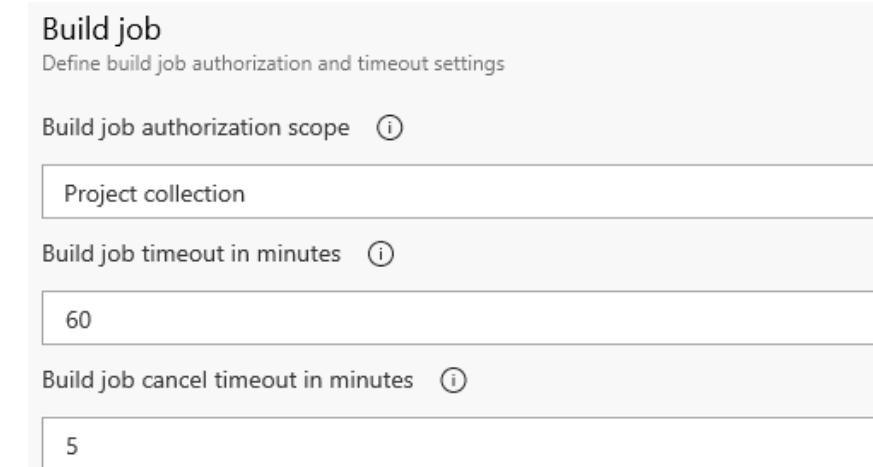
- Specify the authorization scope for a build job.
- Project Collection if the build needs access to multiple projects.
- Current Project if you want to restrict this build to have access only the resources in the current project.

Build job
Define build job authorization and timeout settings

Build job authorization scope Project collection

Build job timeout in minutes 60

Build job cancel timeout in minutes 5



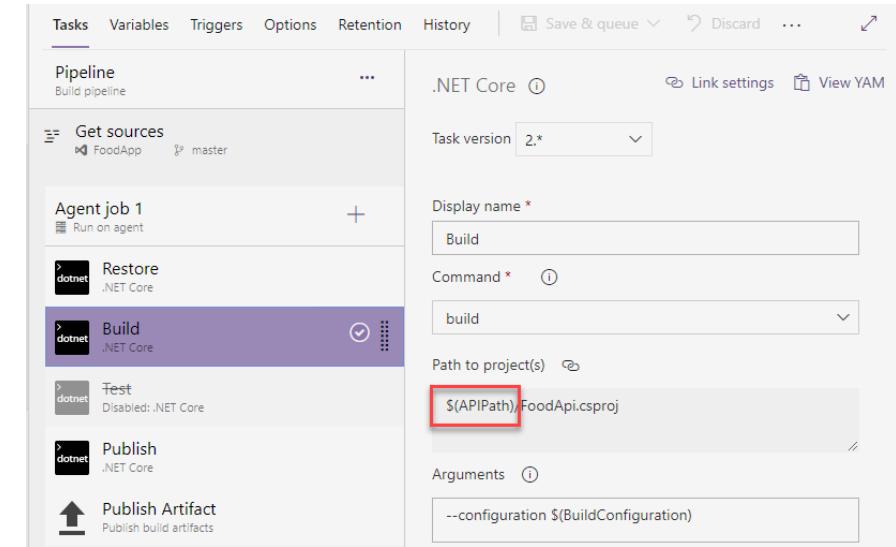
Using Variables

Help to reduce complexity when working with

- Environments and
- Resources

Several Variable Types

- Predefined
- Custom
 - Global / Step specific
- Variable Groups
 - Make Accessible through different pipelines



Name ↑	Value
APIPath	FoodApi
BuildConfiguration	Release
BuildPlatform	any cpu

Build Number Formatting and Build Status

- Build number formatting

Build properties
Define general build pipeline setting

Build number format (i)
\$(date:yyyyMMdd)\$(rev:r)

- Build status (enabled, paused, disabled)

The new build request is processing

- Enabled - queue and start builds when eligible agent(s) available
- Paused - queue new builds but do not start
- Disabled - do not queue new builds

Badges

- Indicate the State of a Build - can be added to GitHub

Status badge

 Azure Pipelines succeeded

Image URL

https://dev.azure.com/trainingsimple/FoodApp/_apis/build/status... 

Image URL for specific branch

https://dev.azure.com/trainingsimple/FoodApp/_apis/build/status... 

Markdown link

`![Build status](https://dev.azure.com/trainingsimple/FoodApp/_a...)` 

Configuring Build Retention

Retention policies are used to configure how long runs and releases are to be retained by the system

The screenshot shows the 'Project Settings' page for a project named 'FoodApp'. A red box highlights the 'Project Settings' tab in the top navigation bar. Another red box highlights the 'Settings' tab under the 'Pipelines' section. On the right, the 'Retention policy' section is displayed, containing the following configuration:

Setting	Value
Days to keep artifacts and attachments	30
Days to keep runs	30
Days to keep pull request runs	10
Number of recent runs to retain per pipeline	3

A warning message in the Retention policy section states: "The artifacts and attachments retention setting is being ignored because the runs retention setting is evaluated first."

Below the retention settings, there is a 'General' section with several toggle switches:

- Disable anonymous access to badges (disabled)
- Limit variables that can be set at queue time (disabled)
- Limit job authorization scope to current project (disabled)
- Publish metadata from pipelines (preview) (disabled)

Service Connections

- Service Connections allow consumption of external ressources such as Azure, GitHub, ...
- Azure Pipelines can automatically build and validate every pull request and commit to your GitHub repository

The screenshot shows the 'Service connections' page in the Azure DevOps interface. The left sidebar is titled 'Project Settings' and lists various project management sections: General, Overview, Teams, Permissions, Notifications, Service hooks, Dashboards, Boards, Project configuration, Team configuration, GitHub connections, Repositories, Cross-repo policies, Pipelines, Agent pools, Parallel jobs, Settings, Test management, Release retention, and Service connections*. The 'Service connections*' link is highlighted with a red box. The main content area is titled 'Service connections' and shows a list of available service types. A search bar at the top says 'Search connection types'. Below it, the 'FoodApp' connection is listed under the 'Service connections' section. On the right, a large list of service types is shown, each with a corresponding icon and name. The 'GitHub' service type is highlighted with a red box. Other listed services include Azure Classic, Azure Repos/Team Foundation Server, Azure Resource Manager, Azure Service Bus, Bitbucket Cloud, Chef, Docker Host, Docker Registry, Generic, GitHub Enterprise Server, Jenkins, Jira, Kubernetes, Maven, NuGet, and Other Git.

Service Type
Azure Classic
Azure Repos/Team Foundation Server
Azure Resource Manager
Azure Service Bus
Bitbucket Cloud
Chef
Docker Host
Docker Registry
Generic
GitHub
Github Enterprise Server
Jenkins
Jira
Kubernetes
Maven
NuGet
Other Git

Lab: Enabling Continuous Integration with Azure Pipelines

In this hands-on lab, you will learn how to configure continuous integration with Azure Pipelines. You will perform the following tasks:

- Creating a basic build pipeline from a template
- Tracking and reviewing a build
- Invoking a continuous integration build

✓ Note that you must have already completed the prerequisite labs in the Welcome section.

Lesson 02: Implementing a Build Strategy



Lesson 2 Overview

- Automated Build Workflows
- Implementing Build Triggers
- Working with Hosted Agents
- Implementing a Hybrid Build Process
- Configuring Agent Demands
- Implementing Multi-Agent Builds
- Build-Related Tooling
- Creating a Jenkins Build Job and Triggering CI

Automated Build Workflows

- Azure DevOps can automate a custom workflow that's as large and complex as you need
- Agile teams normally require more than one type of build
- Builds are typically triggered automatically when code is committed
- Builds can also be scheduled – such as a daily build

Implementing Build Triggers

← ARambazamba.FoodApp

master ▾ ARambazamba/FoodApp / azure-pipelines.yml *

```
1
2 trigger:
3   # Which branch triggers the pipeline
4   branches:
5     include:
6       - master
7       - development
8   # Which path triggers the pipeline -> Monorepo
9   paths:
10    include:
11      - FoodApi/*
```

Parts Unlimited-ASP.NET-CI

Tasks Variables **Triggers** Options Retention History | Save & queue

Continuous integration

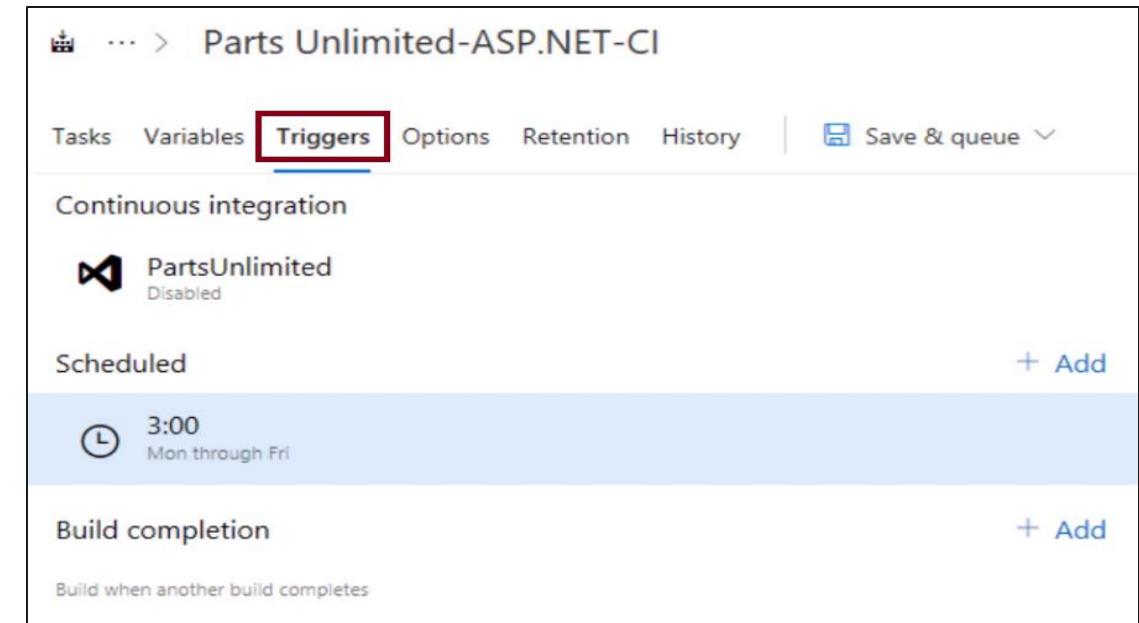
PartsUnlimited Disabled

Scheduled + Add

3:00 Mon through Fri

Build completion + Add

Build when another build completes



Lesson 03: Evaluate Use of Hosted vs Private Agents



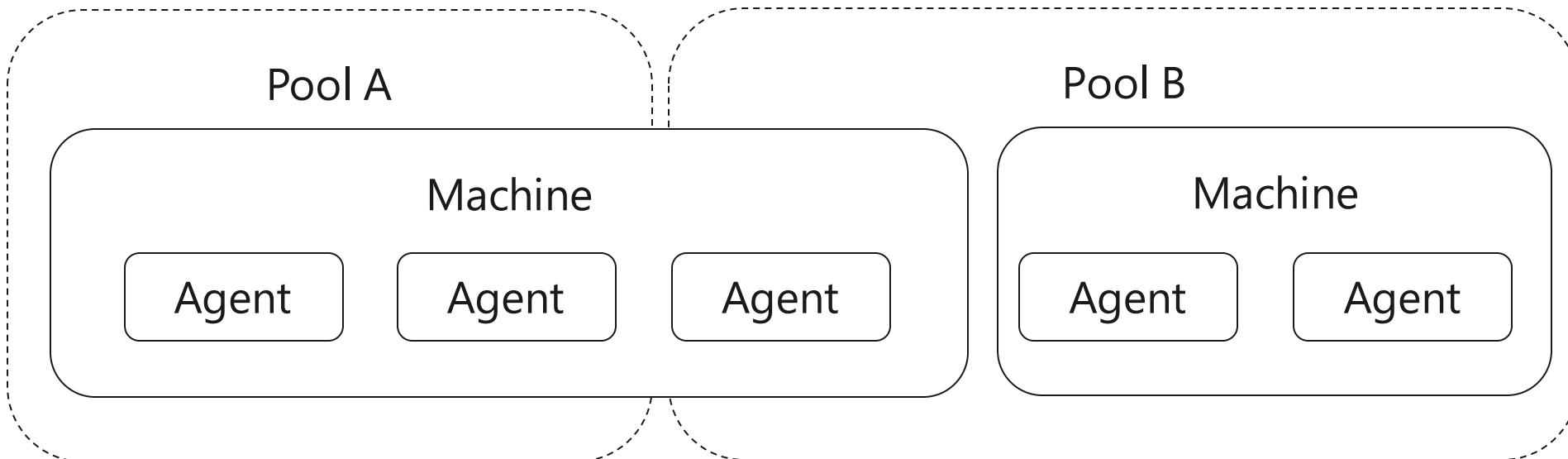
Hosted vs Private Agents

- An agent is installable software that runs one build or deployment job at a time
- Two types of agents:
 - **Microsoft-hosted agents** - Automatically take care of maintenance and upgrades. Each time you run a pipeline, you get a fresh virtual machine. The virtual machine is discarded after one use.
 - **Self-hosted agents** – You take care of maintenance and upgrades. Give you more control to install dependent software needed. You can install the agent on Linux, macOS, Windows machines, or even in a Linux Docker container.

The screenshot shows the DevPool interface with the following details:

- Header:** DevPool, Search bar, and various navigation icons.
- Section:** DevPool (highlighted in red).
- Sub-section:** Agents (selected tab).
- Buttons:** Update all agents and New agent.
- Table Headers:** Name, Last run, Current status, Agent version, Enabled.
- Table Data:** One row for 'I9DEV' with status 'Online', last run '30. Jan.', current status 'Idle', agent version '2.163.1', and an 'Enabled' toggle switch set to '0'.

Agent Pools



- You can organize agents into agent pools - Defines the sharing boundary
- In Azure Pipelines, agent pools are scoped to the Azure DevOps organization; so you can share an agent pool across projects

Working with Hosted Agents

- Azure Pipelines provides a Microsoft-hosted agent pool named Azure Pipelines that offers several virtual machine images to choose from

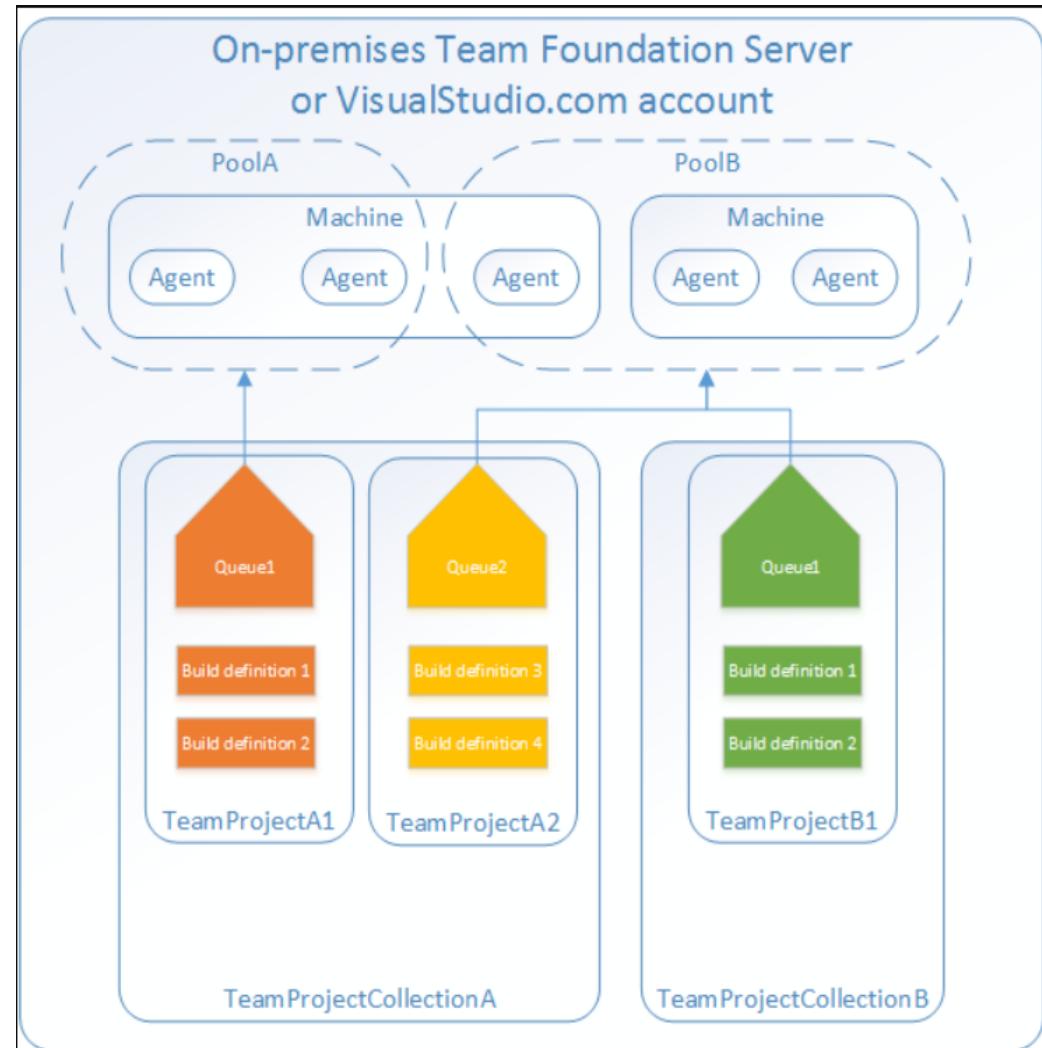
Image	Classic Editor Agent Specification	YAML VM Image Label
Windows Server 2019 with Visual Studio 2019	windows-2019	windows-latest OR windows-2019
Windows Server 2016 with Visual Studio 2017	vs2017-win2016	vs2017-win2016
Ubuntu 18.04	ubuntu-18.04	ubuntu-latest OR ubuntu-18.04
Ubuntu 16.04	ubuntu-16.04	ubuntu-16.04
macOS X Mojave 10.14	macOS-10.14	macOS-10.14
macOS X Catalina 10.15	macOS-10.15	macOS-latest OR macOS-10.15

Azure Pipelines Pipeline Editor screenshot showing the configuration for a build pipeline named "Parts Unlimited-ASP.NET-CI". The pipeline consists of a single job named "Agent job 1" which runs the tasks: Get sources (using Hosted VS2017), Use NuGet 4.4.1, NuGet restore, Build solution, Test Assemblies, Publish symbols path, and Publish Artifact.

The "Agent pool" dropdown menu is open, showing the "Hosted VS2017" option selected. Other options include Hosted, Hosted Linux Preview, Hosted macOS, Hosted Ubuntu 1604, Hosted VS2017, Hosted Windows Container, and Private.

Typical Situations for Agent Pools

- You're a member of a project and you want to use a set of machines owned by your team for running build and deployment jobs
- You're a member of the infrastructure team and would like to set up a pool of agents for use in all projects
- You want to share a set of agent machines with multiple projects, but not all of them



Security of Agent Pools

- Roles are defined on each agent pool, and membership in these roles governs what operations you can perform on an agent pool

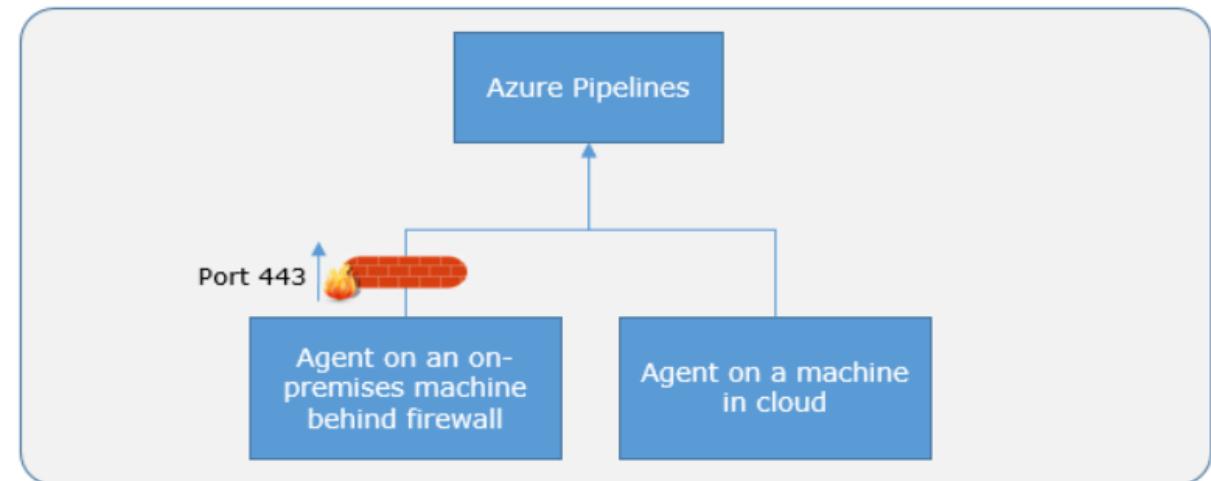
Role	Purpose
Reader	Can view the organization agent pool as well as agents. You typically use this to add operators that are responsible for monitoring the agents and their health.
Service Account	Can use the organization agent pool to create a project agent pool in a project. If you follow the guidelines above for creating new project agent pools, you typically do not have to add any members here.
Administrator	In addition to all the above permissions, members of this role can register or unregister agents from the organization agent pool. They can also refer to the organization agent pool when creating a project agent pool in a project. Finally, they can also manage membership for all roles of the organization agent pool. The user that created the organization agent pool is automatically added to the Administrator role for that pool.

Lesson 08: Setup Private Agents



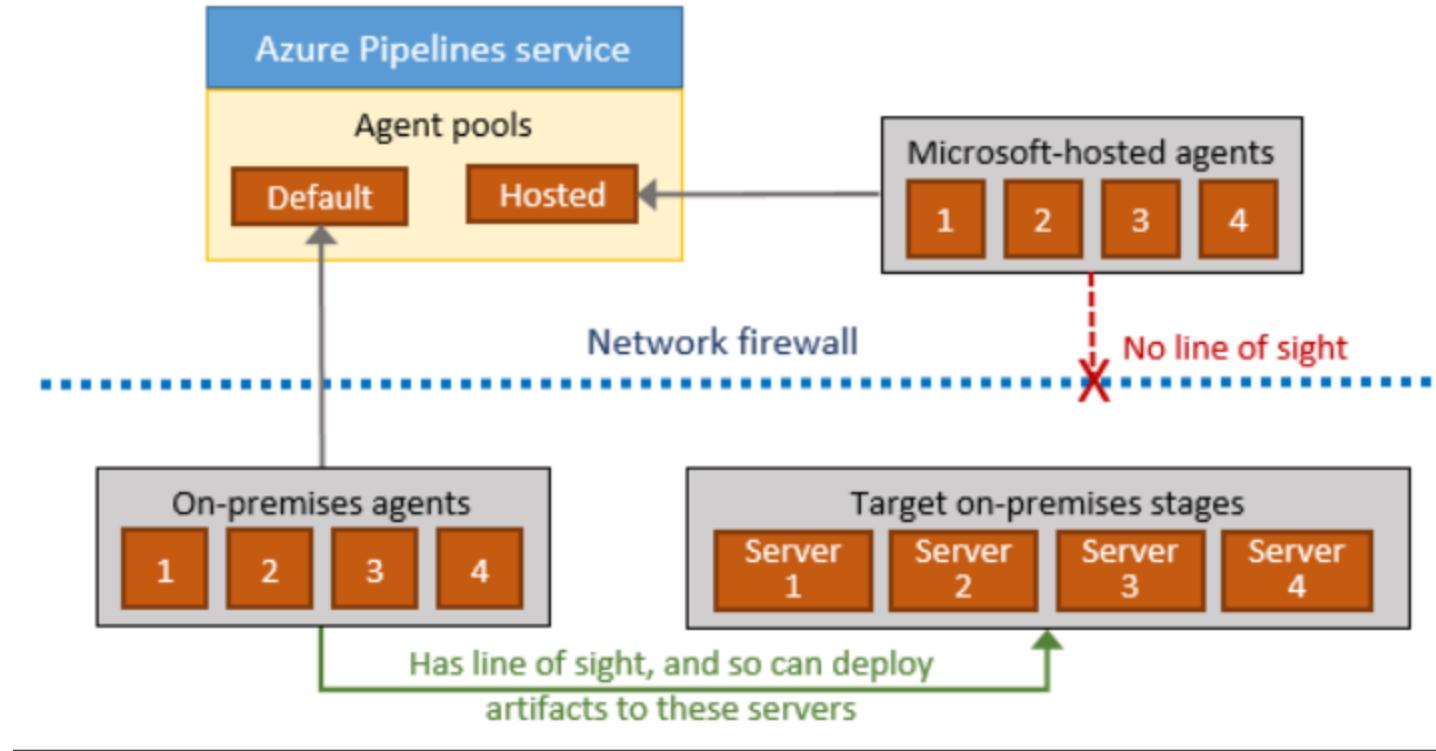
Communication with Azure Pipelines

- The agent determines which job it needs to run, and to report the logs and job status
- Communication is always initiated by the agent
- All the messages from the agent to Azure Pipelines are over HTTPS



Communication to Deploy to Target Servers

- Agent must have "line of sight" connectivity to servers
- Microsoft-hosted agent pools, by default, have connectivity to Azure websites and servers running in Azure
- You may need to manually configure connectivity



Other Considerations

- Authentication
 - To register an agent, you need to be a member of the administrator role in the agent pool
- Personal Access Tokens
 - Generate and use a PAT to connect an agent with Azure Pipelines
- Interactive vs Service processes
- Agent version and upgrades

The screenshot shows the 'Capabilities' tab for an agent named 'I9DEV'. It has two sections: 'User-defined capabilities' and 'System capabilities'. In the 'User-defined capabilities' section, there is a link to 'Add a new capability'. In the 'System capabilities' section, there is a table with three rows:

Name	Value
Agent.Name	I9DEV
Agent.Version	2.160.1
Agent.ComputerName	I9DEV

Implementing a Hybrid (Self Hosted) Build Process

- Agents available for Windows, Linux, macOS
- Docker - Windows / Ubuntu Container
- Can be executed behind proxy using port 8888

The screenshot shows the Azure DevOps interface for managing agent pools. On the left, a sidebar menu highlights 'Agent pools'. The main area displays a list of agent pools:

Name	Queued jobs
Azure Pipelines	Azure Pipelines
Default	Azure Pipelines
DevPool	Alexander Pajer

The 'DevPool' row is selected and highlighted with a red box. The interface then transitions to the 'Agents' tab for the 'DevPool' pool, showing a single agent entry:

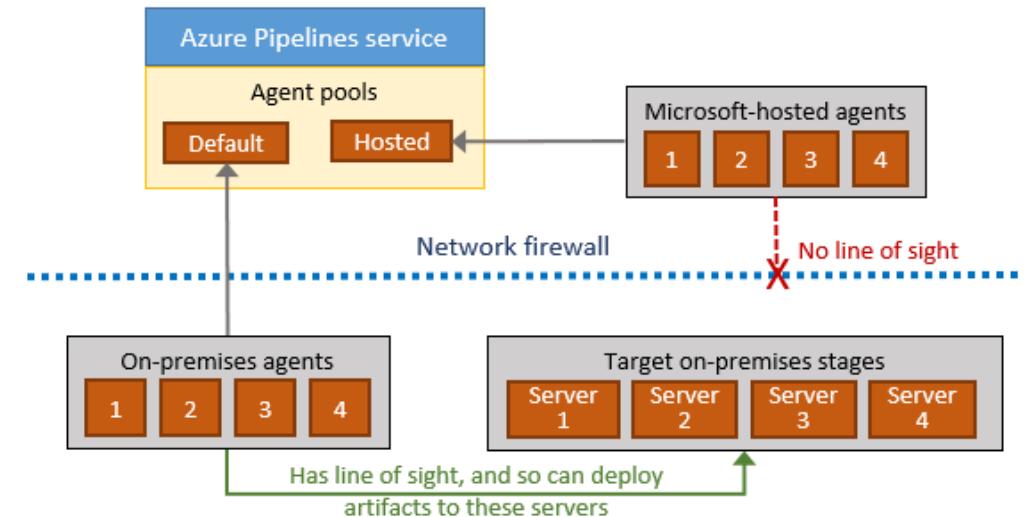
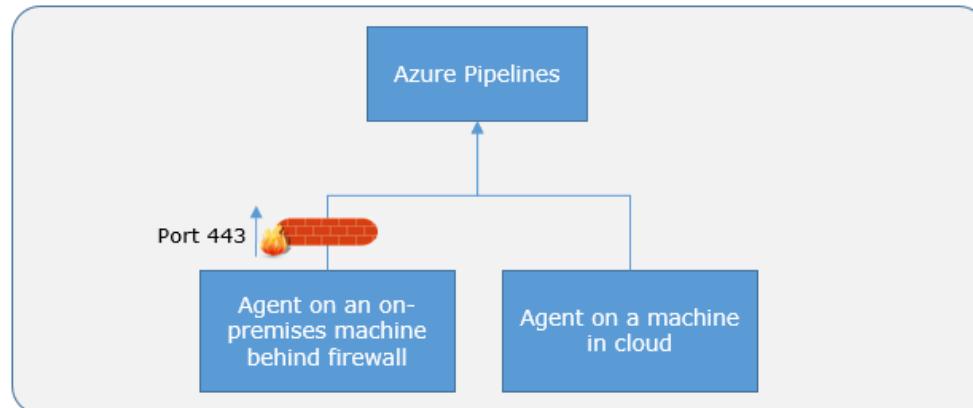
Name	Last run	Current st...	Agent ver...	Ena...
I9DEV	30. Jan.	Idle	2.163.1	<input checked="" type="checkbox"/>

The 'Agents' tab is active, indicated by a blue underline. Buttons for 'Update all agents' and 'New agent' are visible at the top right of this section.

Hybrid Build Security

Why Hybrid?

- Security Issues (On-Prem),
- Special Demands
 - i.e. SharePoint Server Side Build -> Buildtools für Visual Studio



Configuring Agent Demands

Every self-hosted agent has a set of capabilities that indicate what it can do

- User Capabilities
- System Capabilities
- Agents can have different authorization and timeout settings

The screenshot shows the 'Capabilities' tab of an agent configuration page. It includes sections for 'USER CAPABILITIES' and 'SYSTEM CAPABILITIES', each with a table of key-value pairs.

USER CAPABILITIES

Shows information about user-defined capabilities supported by this host

SYSTEM CAPABILITIES

Shows information about the capabilities provided by this host

Capability name	Capability value
Agent.ComputerName	GREGP50
Agent.HomeDirectory	C:\agent
Agent.Name	GREGP50
Agent.OS	Windows_NT
Agent.OSArchitecture	X64
Agent.OSVersion	10.0.17134
Agent.Version	2.141.2
ALLUSERSPROFILE	C:\ProgramData
APPDATA	C:\Users\Greg\AppData\Roaming
AzurePS	5.7.0
bower	C:\Users\Greg\AppData\Roaming\npm\bower.cmd

Implementing Multi-Agent Builds

Adding multiple jobs to a pipeline lets you:

- Break your pipeline into sections that need different agent pools, or self-hosted agents
 - Publish artifacts in one job and consume them in one or more subsequent jobs
 - Build faster by running multiple jobs in parallel
 - Enable conditional execution of tasks
- ✓ You can configure the number of parallel jobs

AZ-400.2

Module 02:

Managing Code Quality and Security Policies

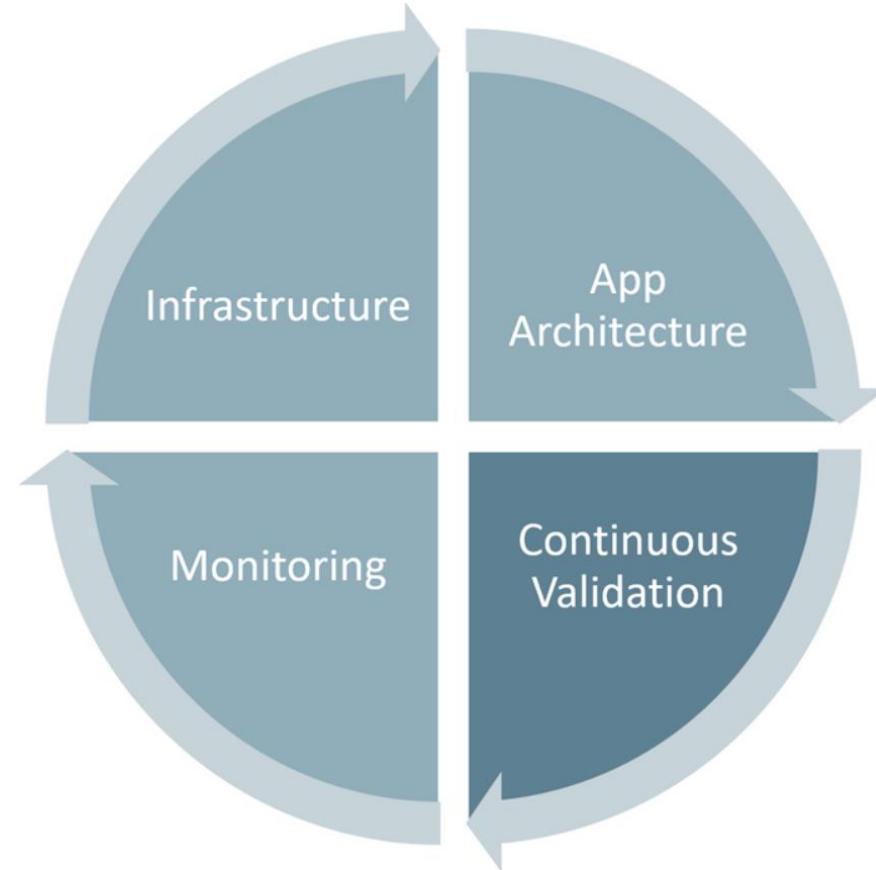


Lesson 01: Introduction to Security



Introduction to Security

- Securing applications is a continuous process that encompasses secure infrastructure, designing an architecture with layered security, continuous security validation, and monitoring for attacks
- Security is everyone's responsibility

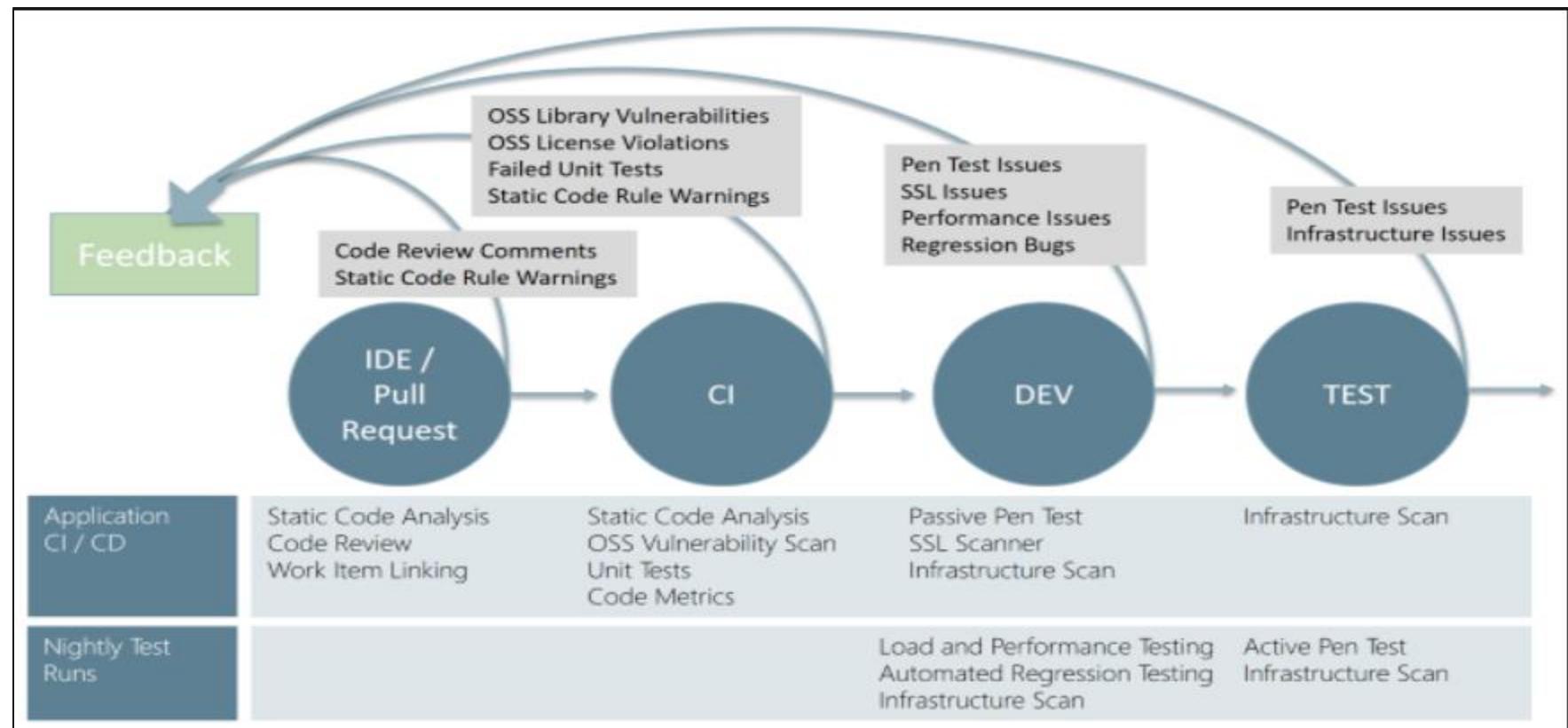


Continuous Integration - Security Testing

- The CI build should be executed as part of the pull request (PR-CI) process and once the merge is complete
- Several tools are available:
 - Visual Studio Code Analysis and the Roslyn Security Analyzers
 - SonarCloud - Bug & Vulnerability Scanner
 - WhiteSource Bolt - Open Source Security Scanner
 - OWASP ZAP - Penetration Testing Tool

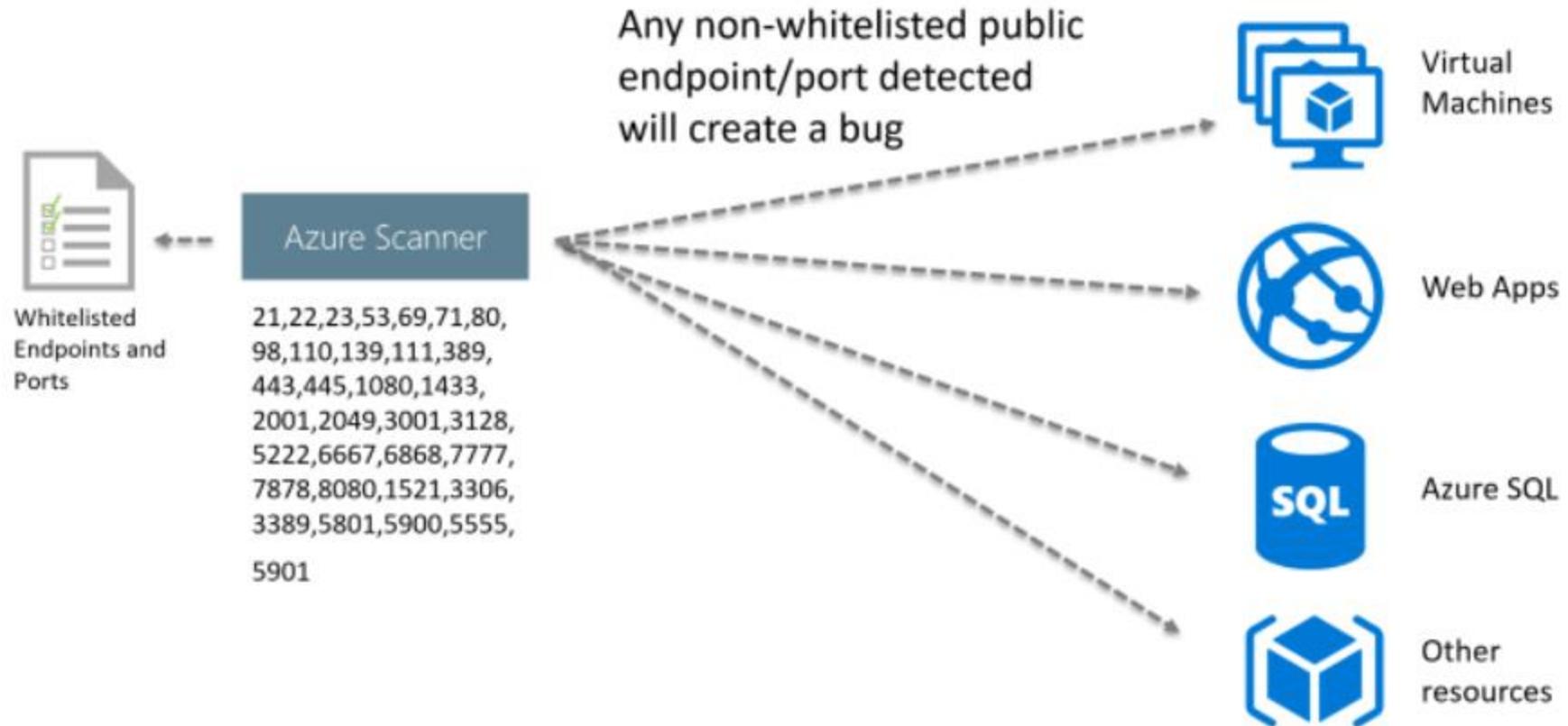
Key Validation Points

- Continuous security validation should be added at each step from development through production



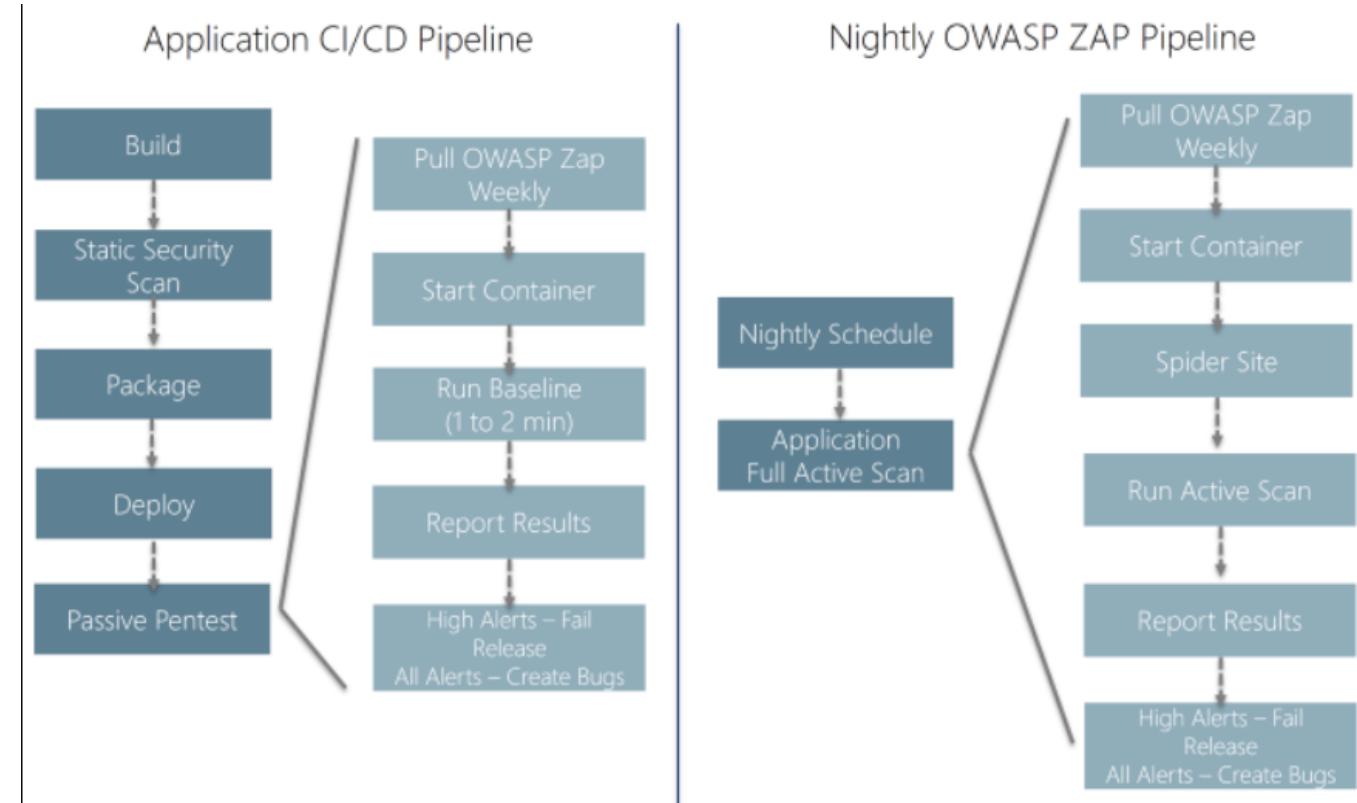
Infrastructure Vulnerabilities

- Be sure to validate the infrastructure
- Use the Azure Security Center and Azure Policies



Application Deployment to DEV and TEST

- OWASP ZAP can be used for penetration testing
- Testing can be active or passive
- Conduct a quick baseline scan to identify vulnerabilities
- Conduct nightly more intensive scans



Results and Bugs

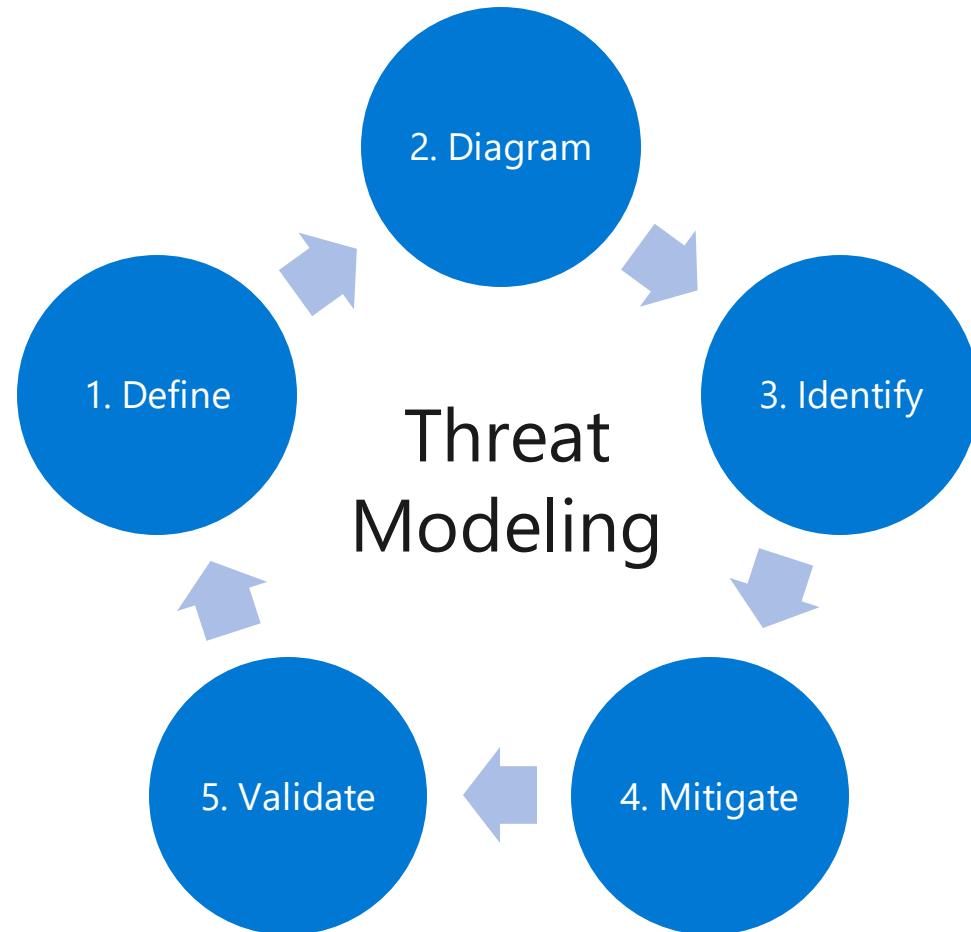
- OWASP ZAP provides a report with results and bugs
- Use a holistic and layered approach to security

The screenshot shows the OWASP ZAP Nightly / Release-40 interface. At the top, there are tabs for Backlog, Board, and Capacity. The Board tab is selected, displaying a Kanban board with three columns: New (61.5 h), Active (11 h), and Unparented. The New column contains five bugs, each with a red border and labeled with a bug ID and a brief description. The Active column contains one bug. Below the board, the interface shows a summary of test results: Total tests: 6, Pass percentage: 0%, Run duration: 0s. It includes buttons for Deploy, Save, and Abandon, and links for Expand all, Collapse all, and Create bug. A 'Test' section lists the bugs found during the test cycle.

Test	Description
0/6 Passed - OWASP ZAP Security Tests	Passed
Incomplete or No Cache-control and Pragma HTTP Header Set	Failed
Cookie No HttpOnly Flag	Failed
Cookie Without Secure Flag	Failed
Web Browser XSS Protection Not Enabled	Failed
X-Content-Type-Options Header Missing	Failed
X-Frame-Options Header Not Set	Failed

Threat Modeling

- Define security requirements
- Create an application diagram
- Identify threats
- Mitigate threats
- Validate that threats have been mitigated



Threat Modeling

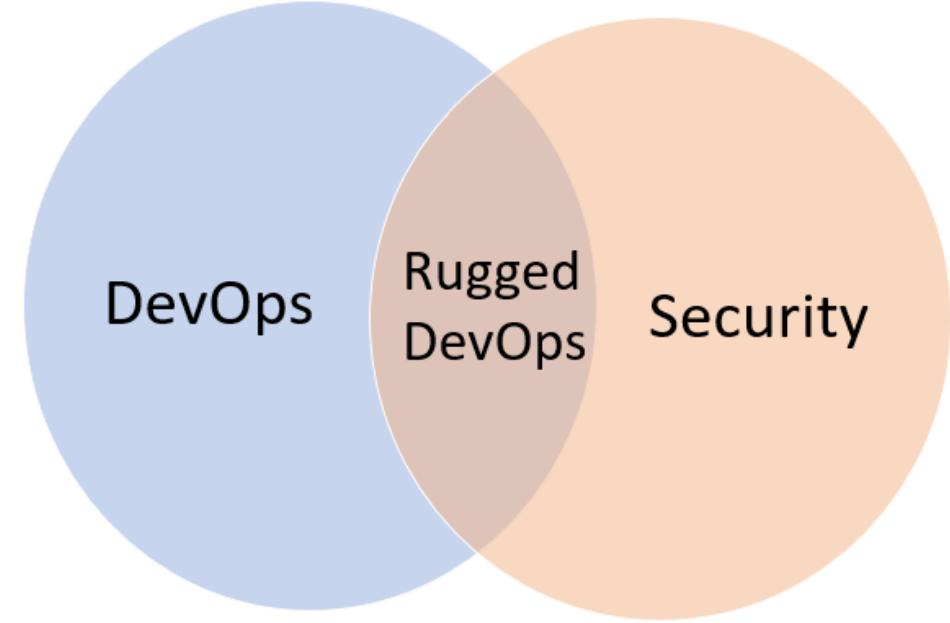


Lesson 05: Implement Tools for Managing Security and Compliance



What is Rugged DevOps

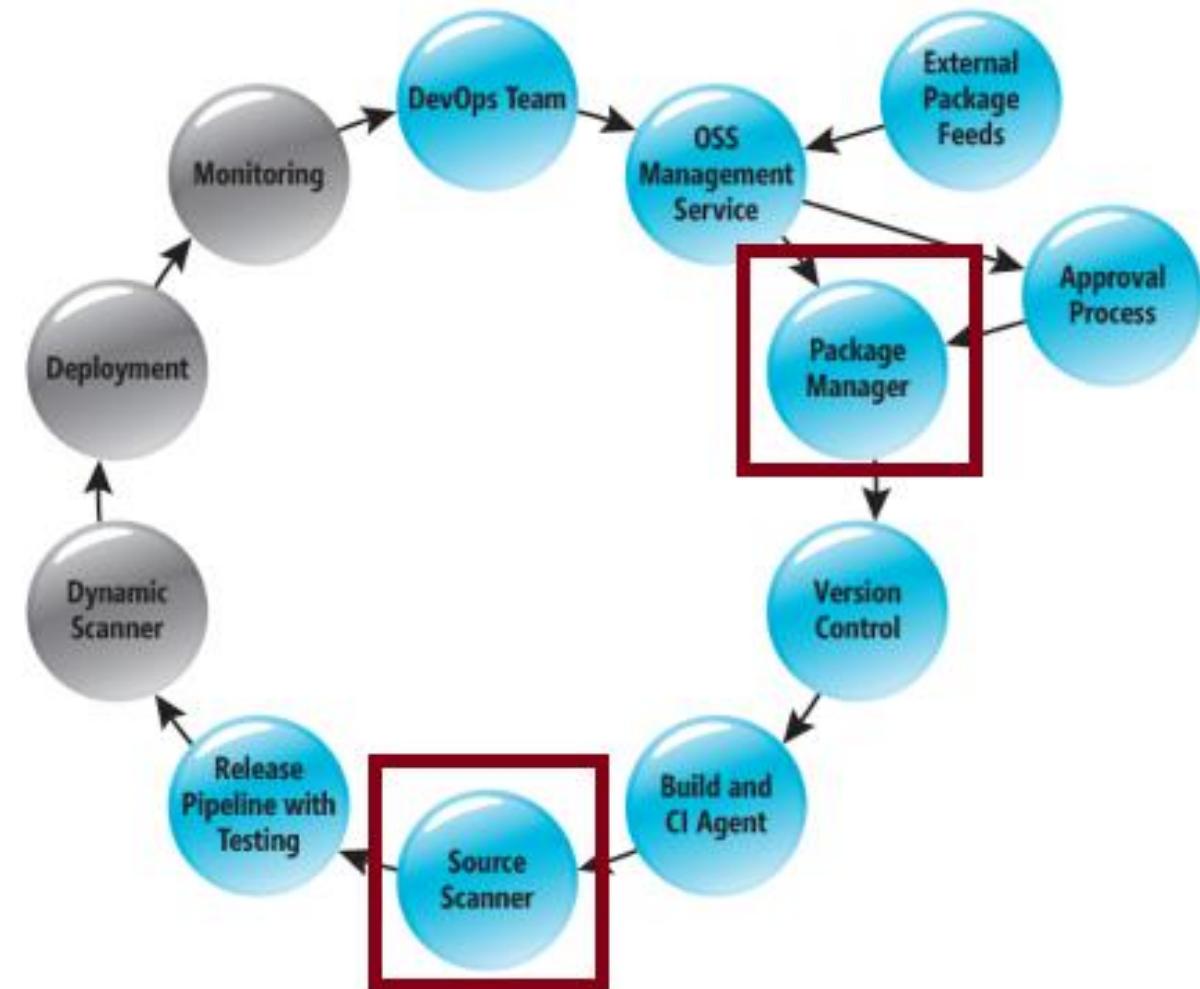
- Rugged DevOps is a set of practices designed to integrate DevOps and security
- The goal is to enable development teams to work fast without introducing unwanted vulnerabilities
- Security strategy includes access control, environment hardening, perimeter protection, and more



Rugged DevOps is also sometimes referred to as *DevOpsSec*

Rugged DevOps pipeline

- Package Management, and the approval process associated with it, accounts for how software packages are added to the pipeline, and the approval process they need to go through
- Source Scanner is for performing a security scan to verify certain security vulnerabilities are not present in our application source code

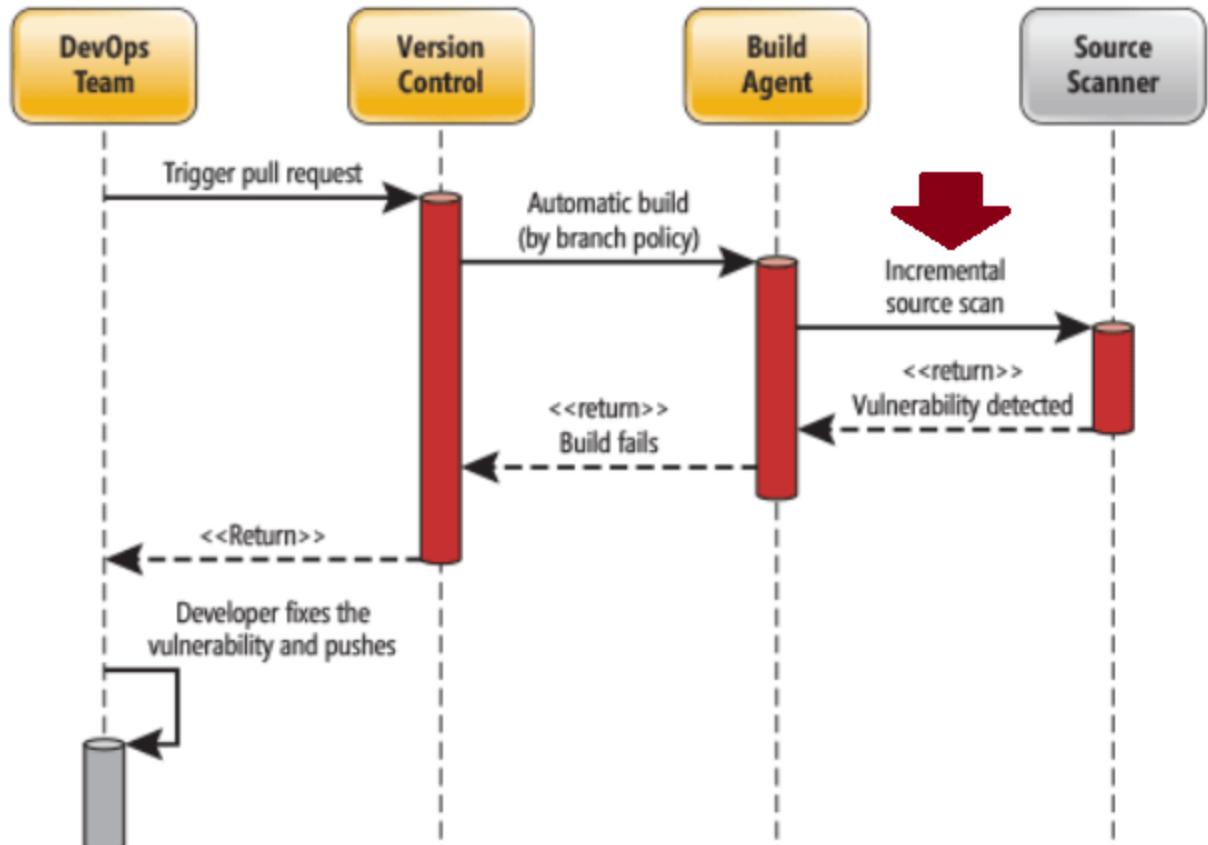


Software composition analysis (SCA)

- Package Management
 - Unique source of binary components
 - Create a local cache of approved components
 - Use Azure Artifacts to share and organize your packages
 - Package types: NuGet, npm, Maven, Gradle, Universal, and Python
- Open Source Software (OSS) Components
 - Reused dependencies can have security vulnerabilities
 - Ensure you have the latest version
 - Check for the correct binaries
 - Promptly address vulnerabilities
 - Analyzing the software to determine its composition can help mitigate potential vulnerabilities

How to integrate software composition analysis Checks into Pipelines

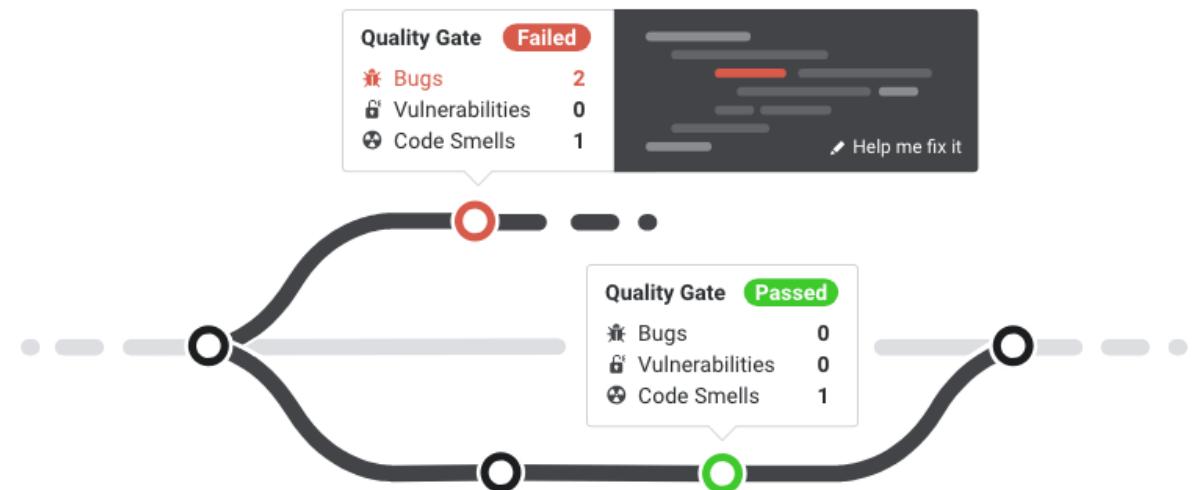
- Pull requests are the way DevOps teams submit changes
- [WhiteSource](#), [Checkmarx](#), [Veracode](#), and [Black Duck by Synopsis](#) can facilitate incremental scans
- Integrate scanning into a team's workflow at multiple points along the path



SonarCloud vs SonarQube

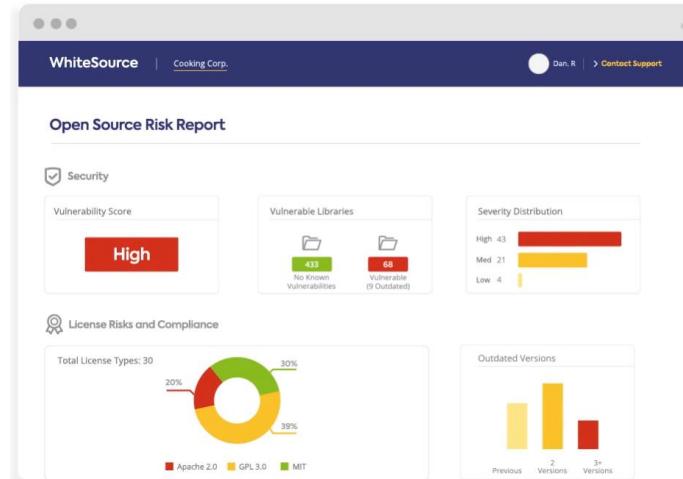
Technical debt – measure between the codebase's current state and an optimal state

- SonarQube is meant to be integrated with on-premise solutions like GitHub Enterprise or BitBucket Server for example
- SonarCloud is meant to be integrated with cloud solutions like Azure Dev Ops or BitBucketCloud



WhiteSource

- Open Source Security Validation Tool
- Find Open Source Vulnerabilities
- Works for:
 - NPM
 - Compiled Libs



	WhiteSource Bolt for Azure DevOps	WhiteSource Full Solution
Languages and Frameworks Coverage	Supports 9 languages (C, C++, GO, JavaScript, PHP, Python, Ruby, Clojure, Swift).	Supports over 200 languages, frameworks, and development environments.
Integrations with DevOps Tools	Integrates with Azure Pipelines.	Integrates with IDEs, package managers, repos, build tools, container registries, CI servers, and AST tools.
Dependency Detection	✗	Fully resolves dependency tree and manifest files, including undeclared dependencies.
Automated Policy Enforcement	✗	Initiate automated workflows based on severity level, license types, library age, and more.
Reporting	Inventory, vulnerabilities, license and outdated components reports per project level only.	Dozens of built-in reports at the project, product or organization level, including: inventory report, due diligence report, risk and attribution reports and even trend reports.

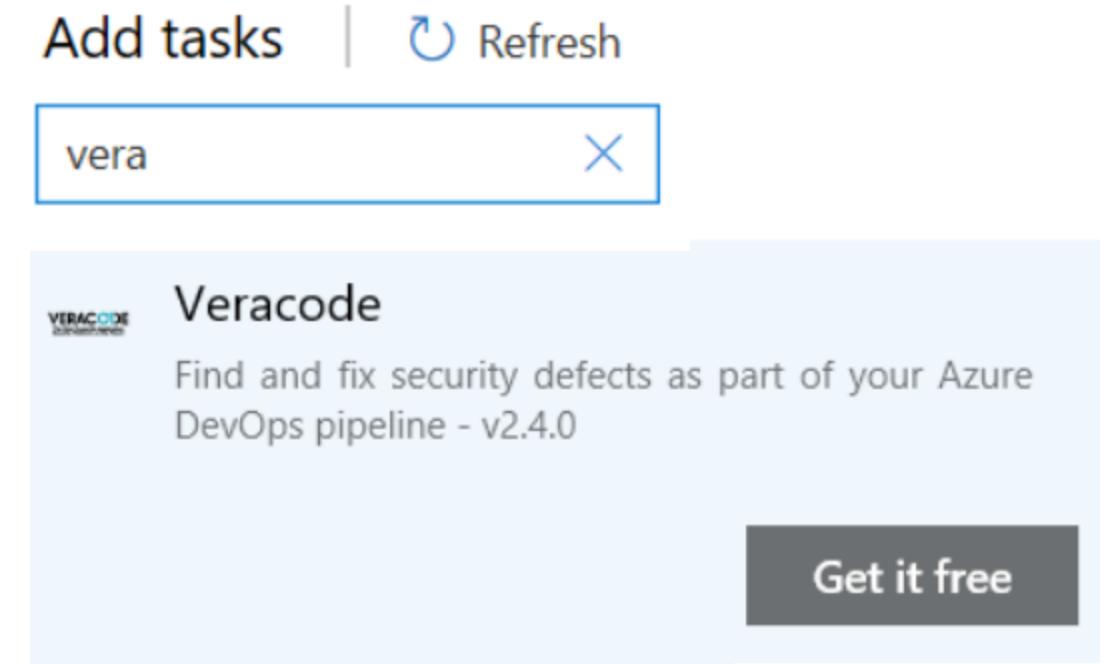
Micro Focus Fortify integration with Azure DevOps pipeline

- Can add build tasks to CI/CD pipeline to help identify vulnerabilities in your source code
- Provides a comprehensive set of software security analyzers
- **Fortify Static Code Analyzer:** Identifies root causes of software security vulnerabilities
- **Fortify on Demand** automatically submits static and dynamic scan requests

All	Build	Utility	Test	Package	Deploy	Tool
	 Fortify on Demand Dynamic Assessment Start Fortify assessment of website					
	 Fortify on Demand Static Assessment Submit code for Fortify on Demand security assessment					
	 Fortify Static Code Analyzer Assessment Run Fortify Static Code Analyzer					
	 Fortify Static Code Analyzer Install Install Fortify SCA on an agent					
	 Fortify WebInspect Dynamic Assessment Run WebInspect dynamic scan					

Veracode integration with Azure DevOps

- Integrate application security into your development tools
- Don't stop for false alarms
- Align your application security practices with your development practices
- Don't just find vulnerabilities, fix them
- Onboarding process allows for scanning on day one

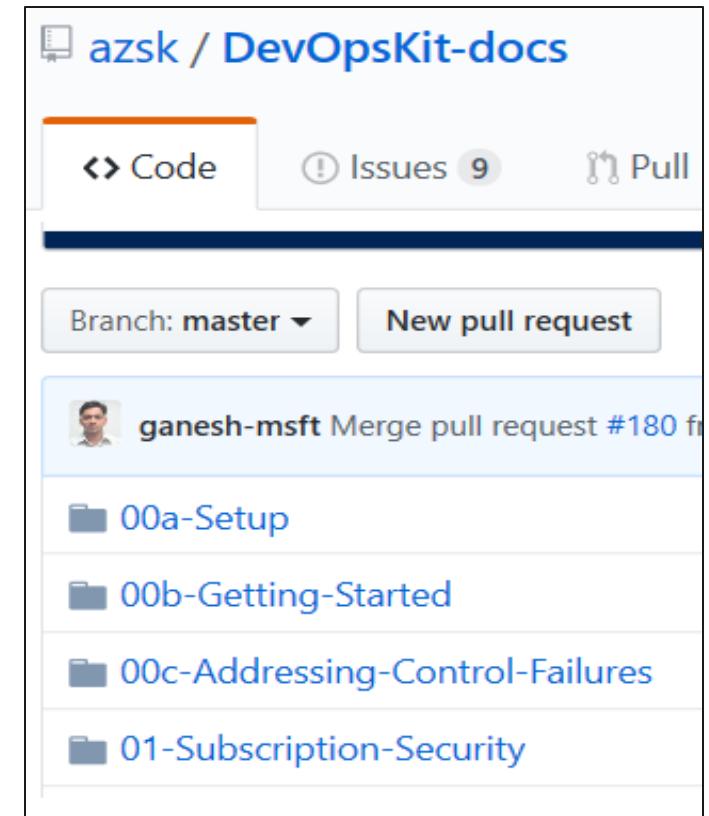


Secure DevOps Kit for Azure (AzSK)

AzSK is a collection of scripts, tools, extensions, automations

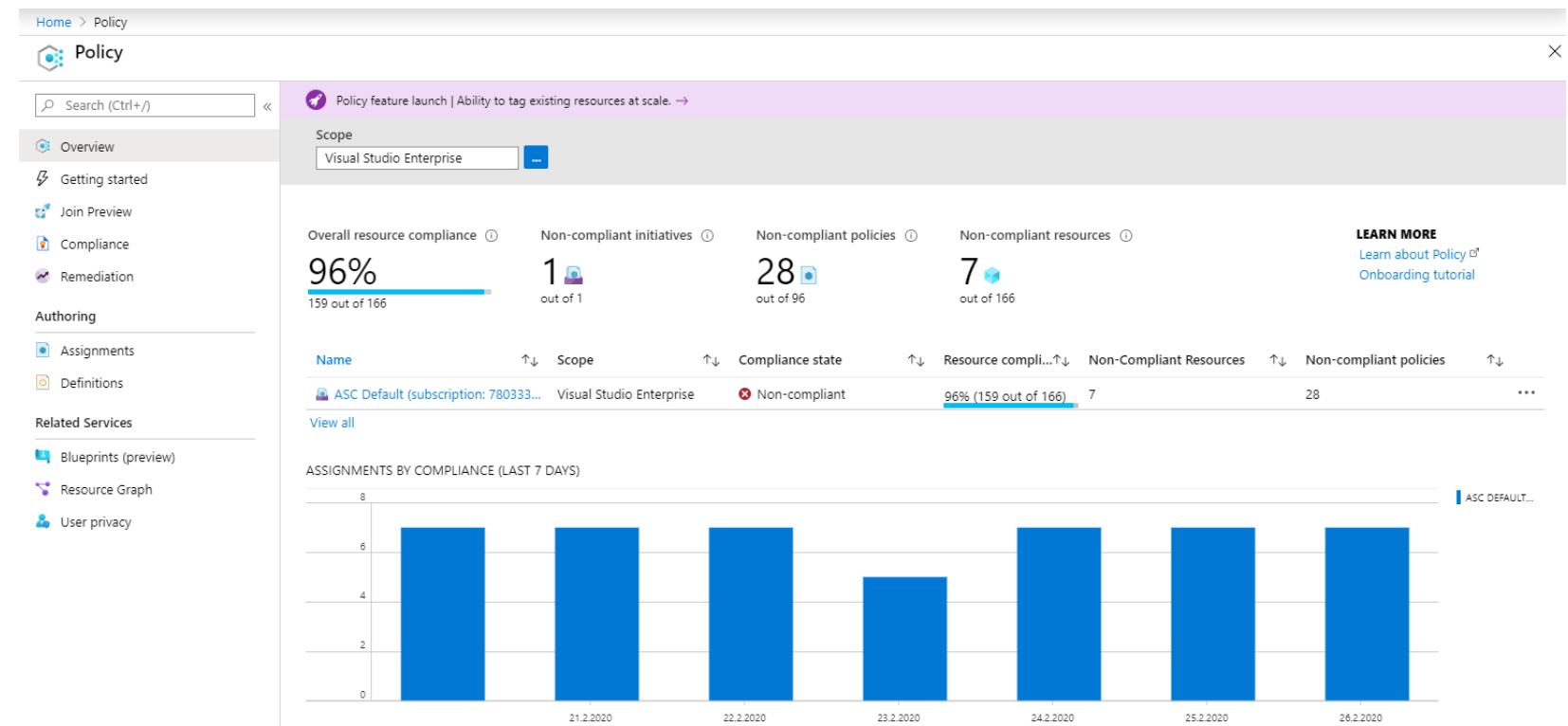
- Helps secure the subscription
- Enables secure development
- Integrates security into CI/CD
- Provides continuous assurance
- Assists with alerts & monitoring
- Governing cloud risks

Available at: <https://github.com/azsk/DevOpsKit-docs>



Azure Policy & Governance

- Enables enforcement restrictions and policies on resources
- Used together with delegated administration



Securing Infrastructure with AzSK

Visual Studio | Marketplace

Azure DevOps > Azure Pipelines > Secure DevOps Kit (AzSK) CICD Extensions for Azure



Secure DevOps Kit (AzSK) CICD Extensions for Azure

Microsoft DevLabs | 990 installs | ★★★★☆ (1) | Free

Collection of extensions that empower DevOps teams to build and deploy applications on Azure with security integrated at every step.

[Get it free](#)

Lab: SonarCloud

In this lab, [Driving continuous quality of your code with SonarCloud](#), you will learn how to integrate Visual Studio Team Services with SonarCloud. You will learn how to:

- Setup a VSTS project and CI build to integrate with SonarCloud
- Analyze SonarCloud reports
- Integrate static analysis into the VSTS pull request process

- ✓ Note that you must have already completed the prerequisite labs in the Welcome section.

Lab: WhiteSource

In this lab, [Managing Open-source security and license with WhiteSource](#), you can use WhiteSource Bolt with Azure DevOps to automatically detect alerts on vulnerable open source components, outdated libraries, and license compliance issues in your code. You will learn how to:

- Detect and remedy vulnerable open source components.
 - Generate comprehensive open source inventory reports per project or build.
 - Enforce open source license compliance, including dependencies' licenses.
 - Identify outdated open source libraries with recommendations to update.
- ✓ Note that you must have already completed the prerequisite labs in the Welcome section.

Lesson 01: Managing Code Quality



Lesson 1 Overview

- Code Quality Defined
- Sources and Impacts of Technical Debt
- Using Automated Testing to Measure and Monitor Technical Debt
- Configuring SonarCloud in a Build Pipeline
- Reviewing SonarCloud Results and Resolving Issues
- Integrating Other Code Quality Tools
- Code Quality Tooling
- Managing Technical Debt with Azure DevOps and SonarCloud

Code Quality Defined

Short deadlines, a lack of coding standards, and poor technical skills can lead to code that is NOT:

- Clear and readable
- Documented
- Efficient
- Maintainable
- Extensible
- Secure

Code Quality Tools

Tool	What it is used for
White Source	Licence Checks
Black Duck	Checkt Open Source Code auf Risiko
Chef	Chef is a powerful automation platform that transforms virtual machine infrastructure on Azure into code.
Octopus Tentacle	Build Pipelines, automated deployment asp.net, Java, node.js on windows, Mac, Linux
Sonar Qube, Sonar Cloud	Inspect sourcecode, bugs, security, > 20 Sprachen
Apache Maven PMD	Code quality checks: Unused variable, empty catch block
Jira	Ticket System
Cobertura	Java: Code coverage testing and publish code
Bullseye coverage	BullseyeCoverage is an advanced C++ code coverage tool used to improve the quality
Coverlet	Coverlet is a cross platform code coverage framework for .NET
JaCoco	JaCoCo - Java Code Coverage Library
SourceGear Vault 10	SourceGear Vault Pro is a version control and bug tracking solution for professional development teams
OWASP ZAP	Owasp Zed Attack Proxy (ZAP): Security tool

Sources and Impacts of Technical Debt

- Technical Debt describes the future penalty that you incur today by making easy or quick choices in software development practices.
- Common sources of technical debt are:
 - Lack of coding style and standards, Lack of or poor design of unit test cases
 - Ignoring or not understanding object orient design principles
 - Monolithic classes and code libraries, Poorly envisioned use of technology, architecture and approach
 - Over-engineering code & Insufficient comments and documentation
 - Not writing self-documenting code
 - Taking shortcuts to meet deadlines
 - Leaving dead code in place

Using Automated Testing to Measure Technical Debt

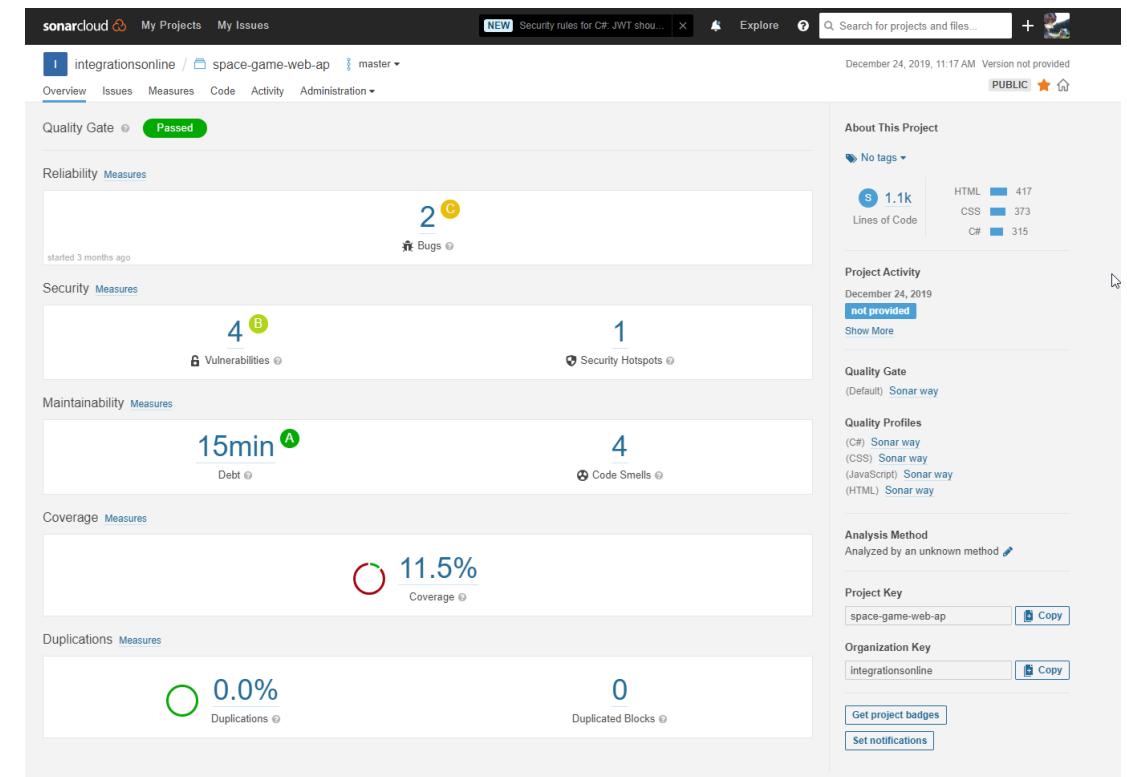
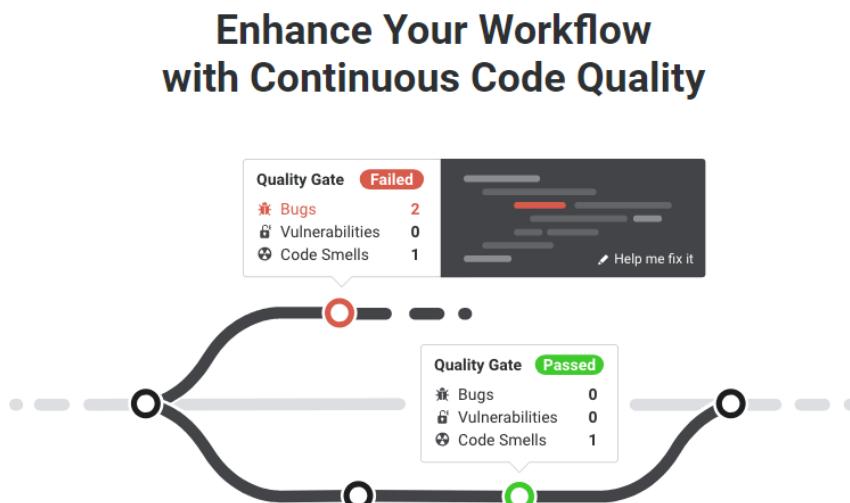
Technical debt:

- Adds problems during development that makes it more difficult to add customer value
 - Saps productivity and frustrates development teams
 - Makes code both hard to understand and fragile
 - Increases the time to make changes, and to validate those changes
 - Starts small and grows over time
- ✓ One way to minimize the accumulation of technical debt, is to use automated testing and assessment

Sonar Cloud

Eliminate bugs and vulnerabilities.

Champion quality code in your projects

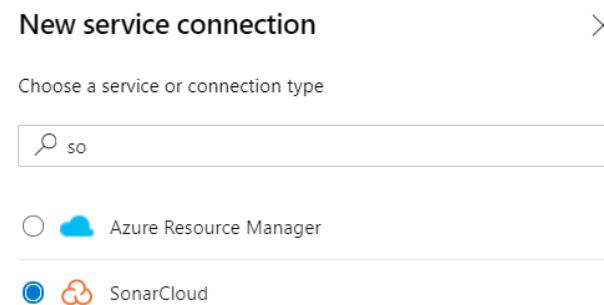


Sonar Cloud Service Connection

Requires:

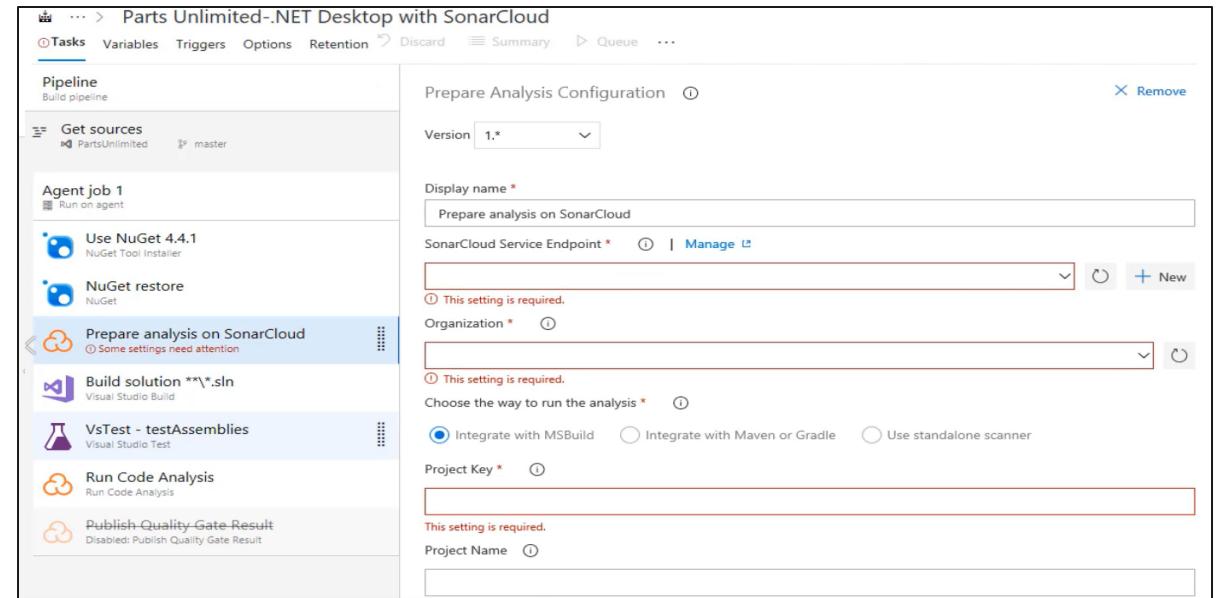
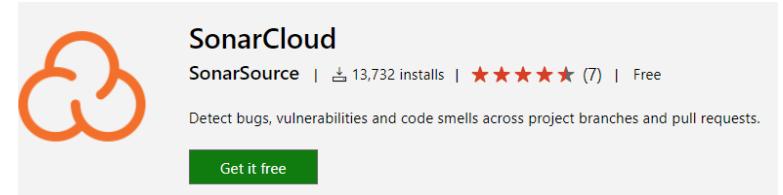
- Login Token generated in Sonarcloud
- Organ Name
- Project Key

The screenshot shows the SonarCloud user interface. At the top, there is a navigation bar with links for 'sonarcloud', 'My Projects', 'My Issues', and a 'NEW Security rules for C#: JWT sh...' card. Below the navigation bar, the user profile 'Alexander Pajer' is displayed, along with 'Profile', 'Security' (which is highlighted with a red box), 'Notifications', and 'Organizations'. A dropdown menu on the right shows 'Alexander Pajer' and 'alexander.pajer@integrations.at', with a sub-menu option 'My Account' also highlighted with a red box. The main content area is titled 'Tokens' and contains a paragraph about using a User Token for security instead of a real login. It includes a 'Generate Tokens' section with a 'Generate' button and a table showing existing tokens. One token listed is 'Analyze "space-game-web-ap"' which was created 'Never'. A 'Revoke' button is visible next to this token entry.



Configuring SonarCloud in a Build Pipeline

- Available as Marketplace Extension
- Requires a Service Connection
- Three Tasks:
 - Prepare Analysis Configuration
 - Run Code Analysis
 - Publish Quality Gate Result
- Typically stored in variables



Reviewing SonarCloud Results and Resolving Issues

The screenshot shows the SonarCloud interface with the 'Issues' tab selected. On the left, there are filters for Type (Bug, Vulnerability, Code Smell, Security Hotspot), Severity (Blocker, Critical, Major), Resolution, Status, and Creation Date. The main area displays a list of issues for the file 'App_Start/BundleConfig.cs' under the project 'PartsUnlimitedWebsite'. There are six issues listed, all of which are 'Code Smell' type, 'Major' severity, and 'Open' status. Each issue has a checkbox, a title, a creation date ('3 years ago'), an effort estimate ('L5', 'L10', 'L14', 'L18', 'L19'), and a 'Comment' link. The last two issues are associated with 'cert' tags.

Type	Count
Bug	55
Vulnerability	3
Code Smell	5.1k
Security Hotspot	2

Severity	Count
Blocker	1
Critical	4.7k
Major	290

Resolution	Count
Info	105

Status	Count
Not assigned	3

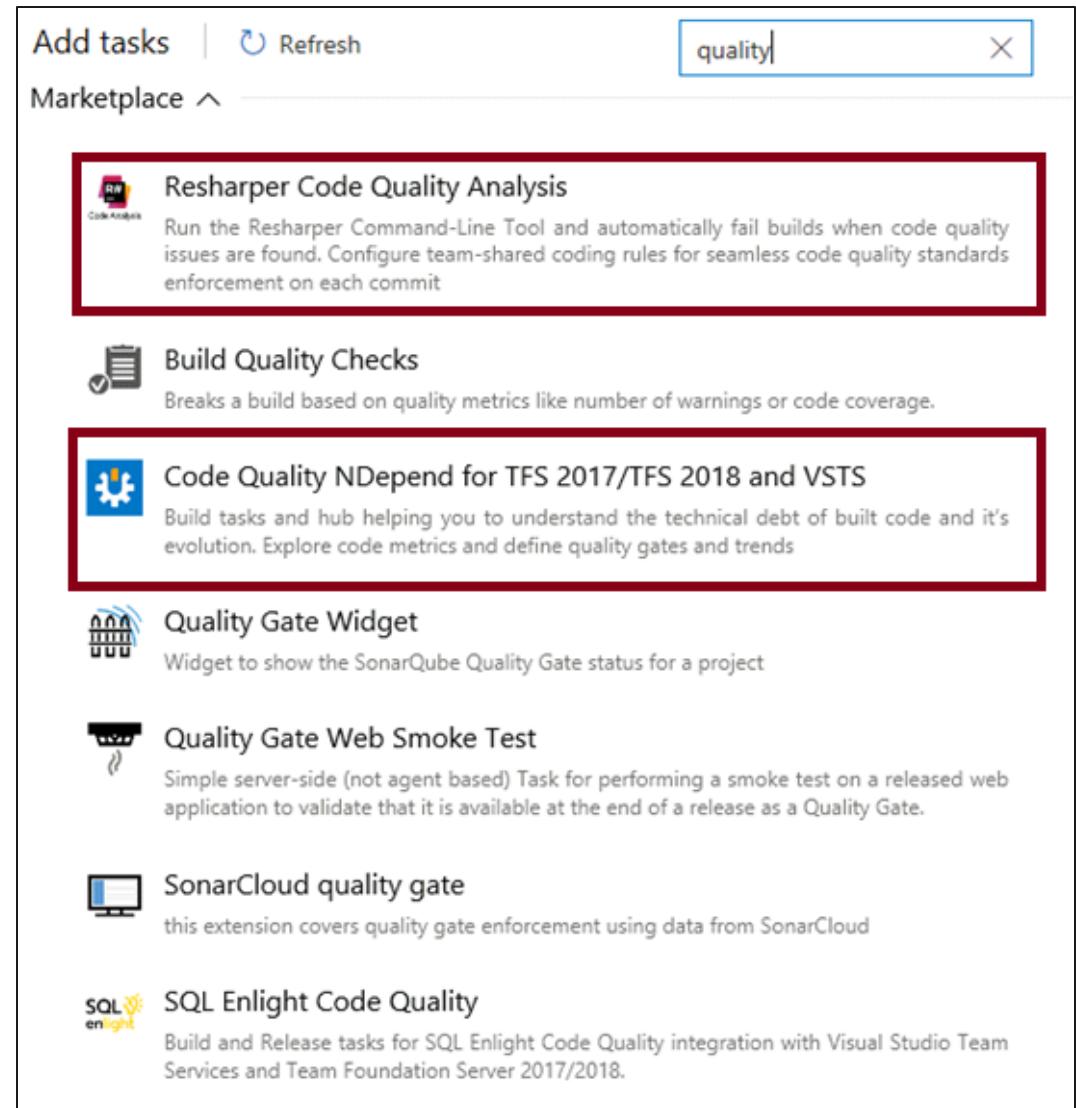
Creation Date	Count
Open	20min effort

Issues List:

- Add a 'protected' constructor or the 'static' keyword to the class declaration. (3 years ago, L5) - Code Smell, Major, Open, Not assigned, 10min effort, Comment, design
- Refactor your code not to use hardcoded absolute paths or URIs. (3 years ago, L10) - Code Smell, Minor, Open, Not assigned, 20min effort, Comment, cert
- Refactor your code not to use hardcoded absolute paths or URIs. (3 years ago, L14) - Code Smell, Minor, Open, Not assigned, 20min effort, Comment, cert
- Refactor your code not to use hardcoded absolute paths or URIs. (3 years ago, L18) - Code Smell, Minor, Open, Not assigned, 20min effort, Comment, cert
- Refactor your code not to use hardcoded absolute paths or URIs. (3 years ago, L19) - Code Smell, Minor, Open, Not assigned, 20min effort, Comment, cert

Integrating Other Code Quality Tools

- NDepend is a Visual Studio extension that assesses the amount of technical debt that a developer has added during a recent development period, typically in the last hour
- Resharper Code Quality Analysis is a command line tool and can be set to automatically fail builds when code quality issues are found



The screenshot shows the Microsoft Marketplace search results for the term "quality". The search bar at the top contains the word "quality". Below the search bar, there is a "Marketplace" button. The results list several items:

- Resharper Code Quality Analysis**: Run the Resharper Command-Line Tool and automatically fail builds when code quality issues are found. Configure team-shared coding rules for seamless code quality standards enforcement on each commit.
- Build Quality Checks**: Breaks a build based on quality metrics like number of warnings or code coverage.
- Code Quality NDepend for TFS 2017/TFS 2018 and VSTS**: Build tasks and hub helping you to understand the technical debt of built code and its evolution. Explore code metrics and define quality gates and trends.
- Quality Gate Widget**: Widget to show the SonarQube Quality Gate status for a project.
- Quality Gate Web Smoke Test**: Simple server-side (not agent based) Task for performing a smoke test on a released web application to validate that it is available at the end of a release as a Quality Gate.
- SonarCloud quality gate**: this extension covers quality gate enforcement using data from SonarCloud
- SQL Enlight Code Quality**: Build and Release tasks for SQL Enlight Code Quality integration with Visual Studio Team Services and Team Foundation Server 2017/2018.

Lab: Managing Technical Debt with Azure DevOps and SonarCloud

In this hands-on lab, you will learn how to manage and report on technical debt using SonarCloud integration with Azure DevOps. You will perform the following tasks:

- Integrate SonarCloud with Azure DevOps and run an analysis
- Analyze the results
- Configure a quality profile to control the rule set used for analyzing your project

✓ Note that you must have already completed the prerequisite labs in the Welcome section.

Lesson 02: Managing Security Policies



Lesson 2 Overview

- Open Source Licensing Challenges
- Avoiding the OWASP Top Ten
- Detecting Open Source Issues with WhiteSource Bolt
- Integrating Other Security Policy Tooling
- Security Policy Tooling
- Checking Vulnerabilities using WhiteSource Bolt and Azure DevOps

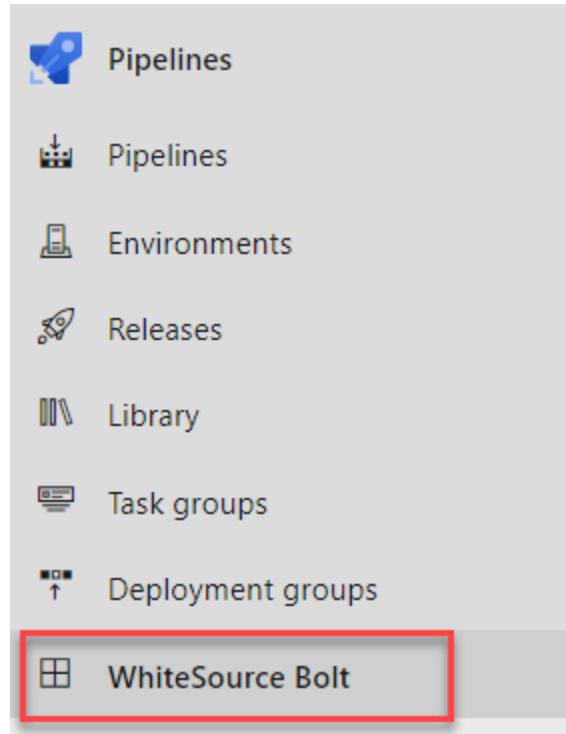
Video: Open Source Licensing Challenges

- Open source software is code that everyone can read, modify, enhance, and share
- Incorporating open source code is convenient but can cause issues:
 - Security
 - Quality
 - Old versions
 - Licensing
- Minimize risk by implementing automated systems to manage the code

Video: Avoiding OWASP Top Ten

1. Injection Attacks
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities
5. Broken Access Control

Detecting Open Source Issues with WhiteSource Bolt



The screenshot shows the 'WhiteSource Bolt Build Report' page for the 'Parts Unlimited-CI' pipeline. The top navigation bar includes 'Logs', 'Summary', 'Tests', 'WhiteSource Bolt Build Report' (which is selected and underlined), 'Release', 'Edit', 'Queue', and more. The main content area is titled 'Parts Unlimited-CI 9' and shows the following data:

- Security**:
 - Vulnerability Score: MEDIUM
 - Vulnerable Libraries: 29 No Known Vulnerabilities, 1 Vulnerable (0 Outdated)
 - Severity Distribution: High 0, Med 1, Low 0. 1 Vulnerable Libraries
 - Aging Vulnerable Libraries: 1 > 90 Days, 0 < 90 Days, 0 < 30 Days
- Security Vulnerabilities (1)**:

Vulnerability	Library	Description	Top Fix
Medium	6.5 bootstrap-3.3.7-3.3.7.js	XSS in data-target in bootstrap (3.3.7 and before)	Replace or update the following file: alert.js, carousel.js, collapse.js, dropdown.js, modal.js https://github.com/twbs/bootstrap/commit/d9be1da55bf0f94a81e8a2c9acf5574fb801306e

```
- task: WhiteSource Bolt@20
  displayName: 'Run WhiteSource Bolt'
```

Integrating Other Security Policy Tooling

- Micro Focus Fortify searches for violations of security-specific coding rules and guidelines
- Checkmarx CxSAST is designed for identifying, tracking and fixing technical and logical security flaws
- BinSkim is a static analysis tool that scans binary files
- OWASP Zed Attack Proxy Scan is an open-source web application for professional penetration testers

Lab: Checking Vulnerabilities using WhiteSource Bolt with Visual Studio Team Services

In this hands-on lab, you will learn how to check for open source vulnerabilities using WhiteSource Bolt in conjunction with Azure DevOps. You will learn how to:

- Integrate WhiteSource Bolt with your Azure DevOps build process
- Detect and remedy vulnerable open source components
- Generate comprehensive open source inventory reports per project or build
- Enforce open source license compliance, including licenses for dependencies
- Identify outdated open source libraries with recommendations to update

 Note that you must have already completed the prerequisite labs in the Welcome section.

Module 2: Review Questions

1. You want to run a penetration test against your application. Which tool could you use?
2. What is code smells? Give an example of a code smell.
3. You are using Azure Repos for your application source code repository. You want to create an audit of open source libraries that you have used. Which tool could you use?
4. Name three attributes of high-quality code.
5. You are using Azure Repos for your application source code repository. You want to perform code quality checks. Which tool could you use?

AZ-400.2

Module 03:

Implementing a Container Build Strategy



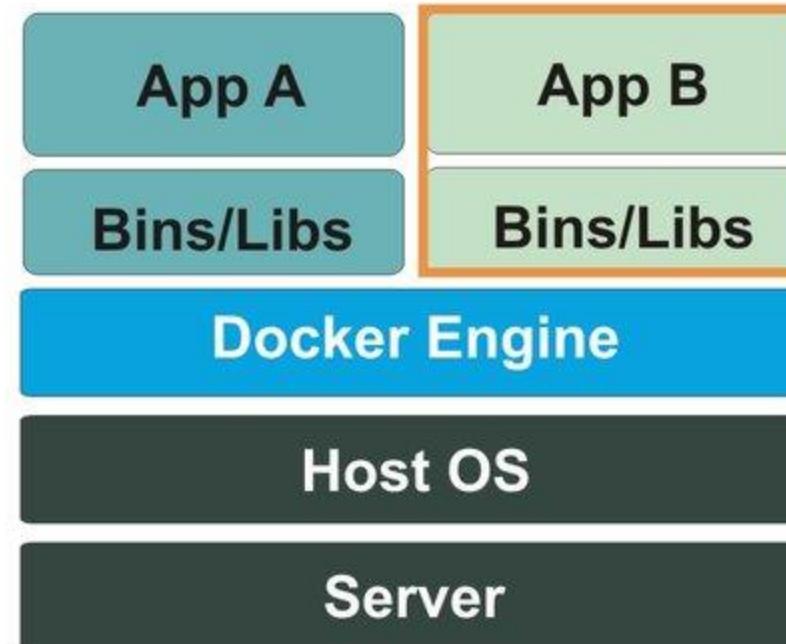
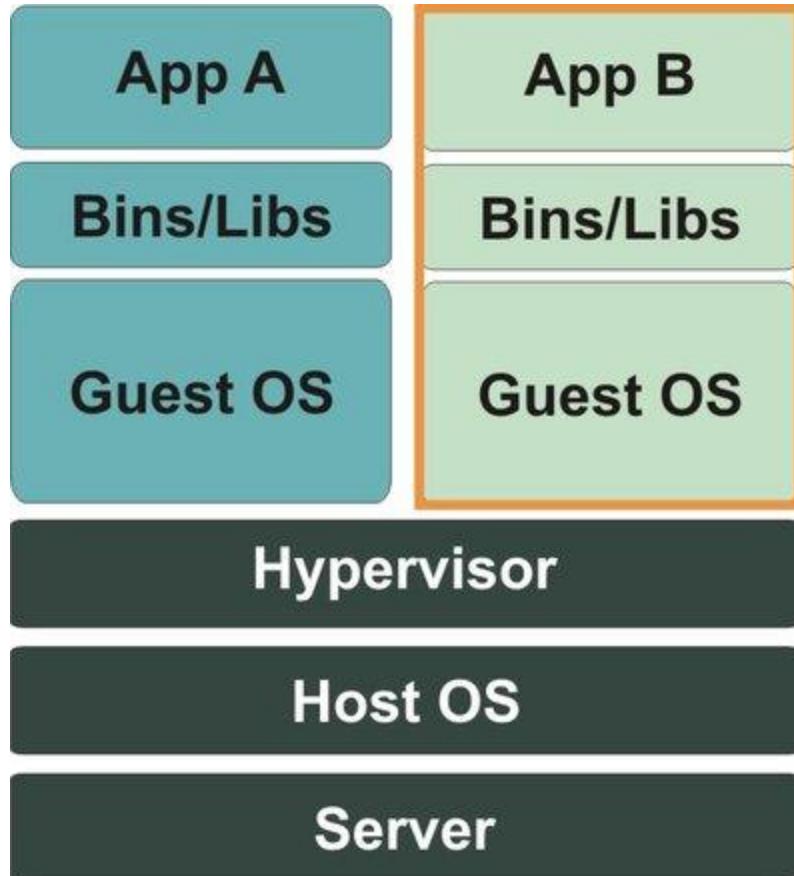
Lesson 01: Implementing a Container Build Strategy



Lesson 1 Overview

- Overview of Containers
- Containers vs Virtual Machines
- Docker Containers and Development
- Microservices and Containers
- Azure Container-Related Services
- Dockerfile Core Concepts
- Creating Multi-Stage Builds
- Creating an Azure Container Registry
- Adding Docker Support to an Existing Application
- Additional Container-Related Resources
- Modernizing an Existing .NET Application with Azure and Docker Images

Virtual Machines vs Containers



Discussion: Containers vs Virtual Machines

In your development environment,

- Do you currently use virtualization of any type?
- Do you prefer to deploy containers or virtual machines?

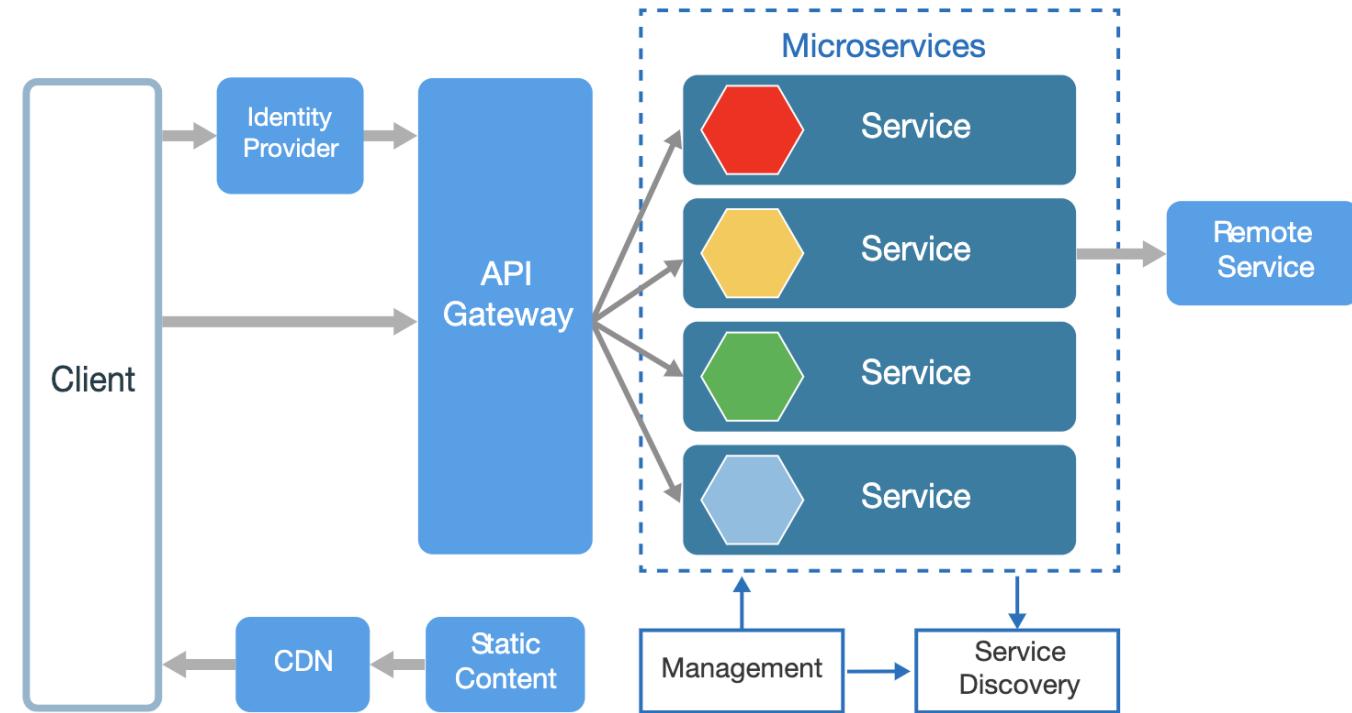
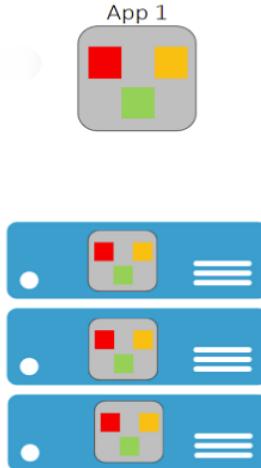
Docker Containers and Development

- Docker is a software containerization platform with a common toolset, packaging model, and deployment mechanism
- Docker greatly simplifies containerization and distribution of applications that can be run anywhere
- A Docker image can be created that will deploy identically across any environment in seconds
- DockerHub has more than 180,000 applications in the public community repository
- Docker organized the Open Container Initiative (OCI)

Microservice Architecture

With Microservices every part of the application is deployed as a fully self-contained component

Monolithic application approach



Azure Container Related Services

- Storing the Image:
 - Azure Container Registry lets you store and manage container images in a central registry
- Hosting
 - Azure Container Instances let you focus on creating your applications rather than provisioning and management of the infrastructure
- Azure Kubernetes Service is the de facto standard for container orchestration
- Azure App Service provides a managed service for both Windows and Linux based web applications

Dockerfile Core Concepts

- Text files that contain the commands needed by docker build to assemble an image
- Repeat the steps that are used to build the application
- Uses Keywords like:
 - FROM
 - WORKDIR
 - RUN
 - COPY
 - ENTRYPOINT
 - EXPOSE

```
FROM mcr.microsoft.com/dotnet/core/sdk:3.1
WORKDIR /app

COPY *.csproj ./
RUN dotnet restore

COPY . ./
RUN dotnet publish -c Release -o out
ENTRYPOINT ["dotnet", "out/SkillsApi.dll"]
EXPOSE 8080/tcp
ENV ASPNETCORE_URLS https://*:5000
```

Multiple Stage Builds

- Keep the image size as small as possible
- Layers are additional instructions added to the Dockerfile
- Multi-stage builds helps optimize the files, improves their readability, and makes them easier to maintain
- Each FROM instruction starts a new stage
- The stages are numbered in order, starting with stage 0
- Stages are named using an AS clause
- Naming stages lets you build them separately

Multistage Build

Uses 2 images

- Build Image with SDK (build)
- Runtime Image (base)

```
FROM mcr.microsoft.com/dotnet/core/aspnet:3.1 AS base
WORKDIR /app
EXPOSE 8080/tcp
ENV ASPNETCORE_URLS https://*:5000

FROM mcr.microsoft.com/dotnet/core/sdk:3.1 AS build
WORKDIR /src
COPY ["*.csproj", ".."]
RUN dotnet restore "SkillsApi.csproj"
COPY . .
RUN dotnet build "SkillsApi.csproj" -c Release -o /app

FROM build AS publish
RUN dotnet publish "SkillsApi.csproj" -c Release -o /app
FROM base AS final
WORKDIR /app
COPY --from=publish /app .
ENTRYPOINT ["dotnet", "SkillsApi.dll"]
```

Create an Azure Container Registry

Container Registry

Microsoft



Azure Container Registry is a private registry for hosting container images. Using the Azure Container Registry, you can store Docker-formatted images for all types of container deployments. Azure Container Registry integrates well with orchestrators hosted in Azure Container Service, including Docker Swarm, DC/OS, and Kubernetes. Users can benefit from using familiar tooling capable of working with the open source Docker Registry v2.



Use Azure Container Registry to:

- Store and manage container images across all types of Azure deployments
- Use familiar, open-source Docker command line interface (CLI) tools
- Keep container images near deployments to reduce latency and costs
- Simplify registry access management with Azure Active Directory
- Maintain Windows and Linux container images in a single Docker registry

Lab: Existing .NET Applications with Azure and Docker Images

In this hands-on lab, you will learn how to modernize an existing ASP.NET application with migration to Docker images managed by the Azure Container Registry. You will learn how to:

- Migrate the LocalDB to SQL Server in Azure
- Using the Docker tools in Visual Studio 2017, add Docker support for the application
- Publish Docker Images to Azure Container Registry (ACR)
- Push the new Docker images from ACR to Azure Container Instances (ACI)

Module 3: Review Questions

1. You are reviewing an existing Dockerfile. How would you know if it's a multi-stage Dockerfile?
2. You are designing a multi-stage Dockerfile. How can one stage refer to another stage within the Dockerfile?
3. What is the line continuation character in Dockerfiles?
4. You are using Azure to manage your containers. Which container orchestration styles are supported?
5. When the Open Container Initiative defined a standard container image file format, which format did they choose as a starting point?