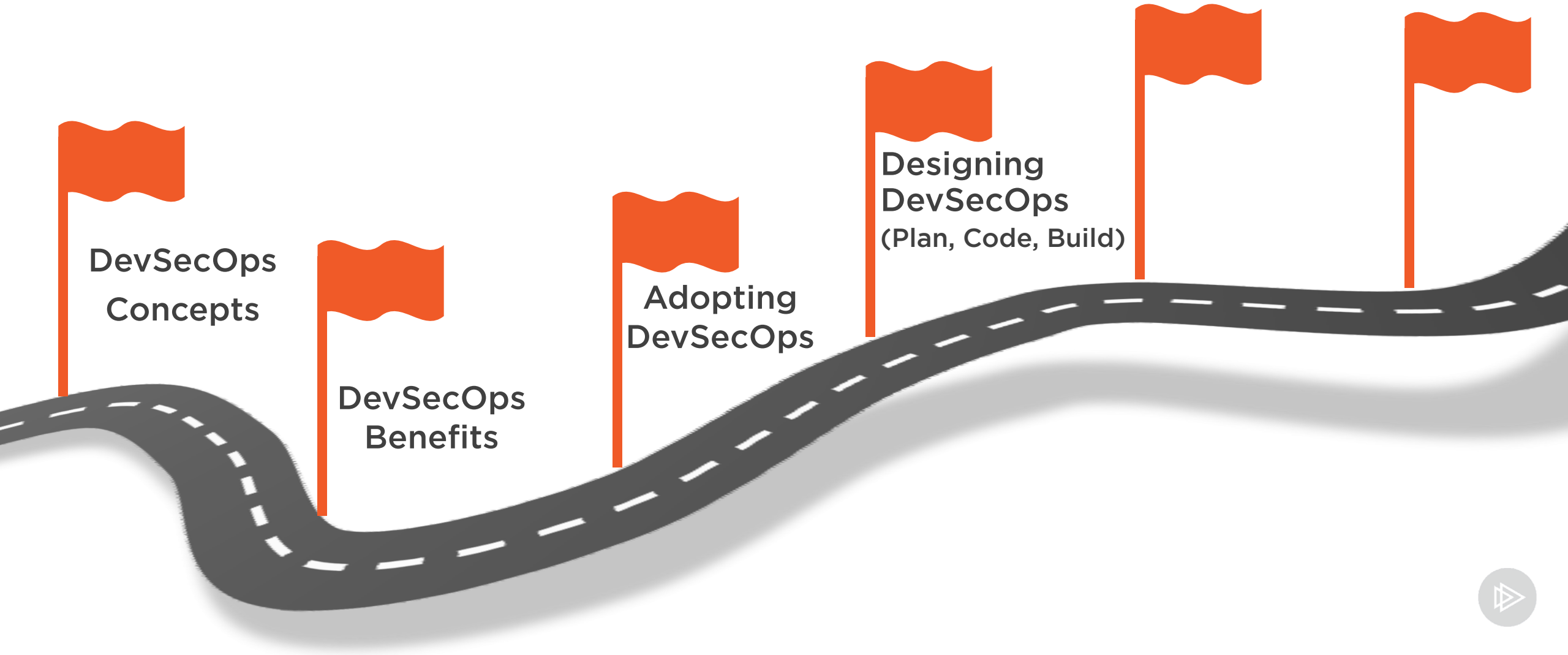
Designing DevSecOps for Plan, Code, and Build SDLC phases

Richard Harpur
INFORMATION SECURITY PROFESSIONAL, CISM

@rharpur www.richardharpur.com

Continue Our DevSecOps Journey

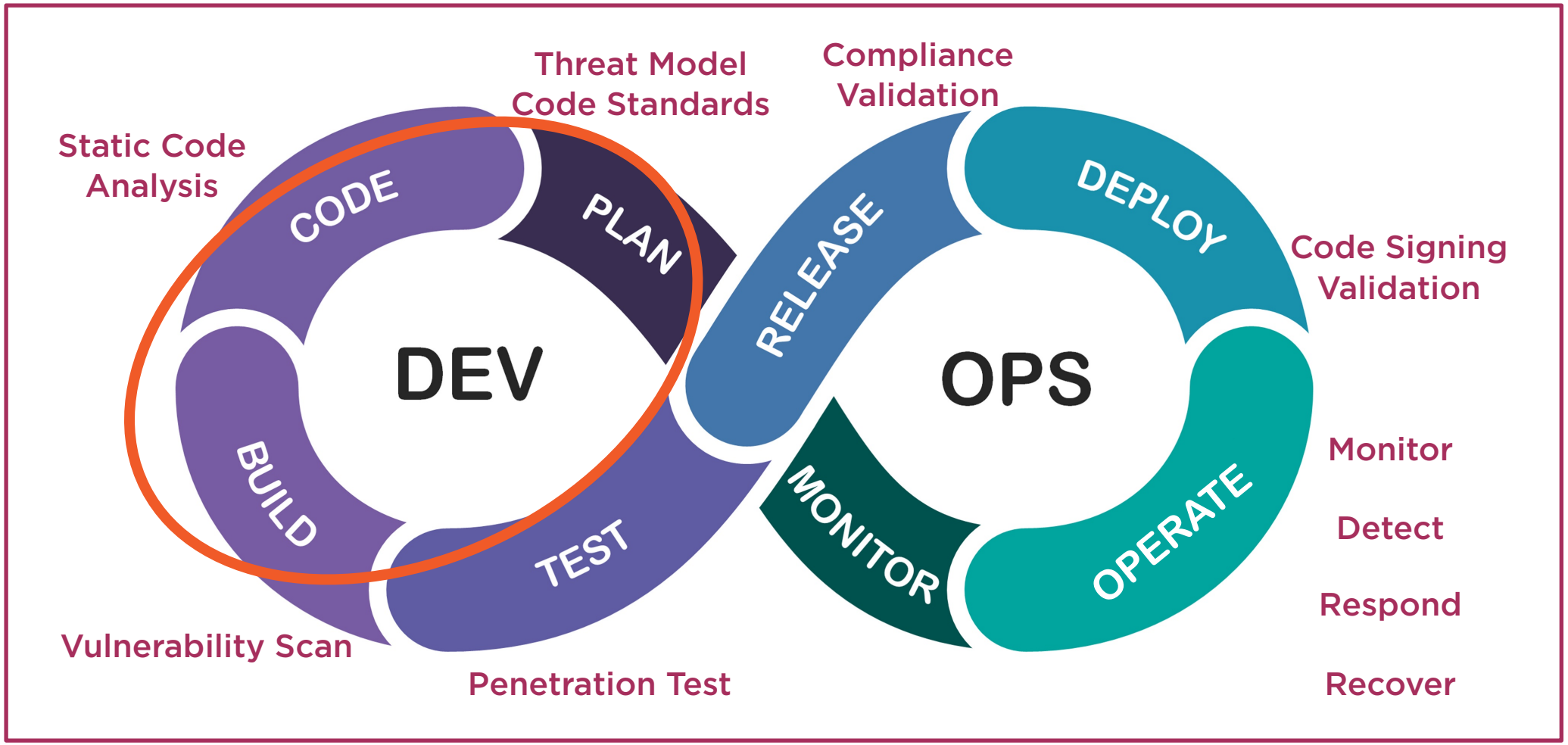


Overview

DevSecOps Requirements for:

- Plan
- Code
- Build

Positioning DevSecOps in Your Lifecycle



Threat Model

"Threat modeling is a process by which potential threats, such as structural vulnerabilities, can be identified, enumerated, and prioritized – all from a hypothetical attacker's point of view"

Source: https://en.Wikipedia.org/wiki/Threat_model

Threat Modelling

- **S** Spoofing
- **T** Tampering
- **R** Repudiation
- **I** Information disclosure / leakage
- **D** Denial of service
- **E** Elevation of privilege

Source: en.wikipedia.org/wiki/STRIDE_(security)

Threat Modelling

S	Spoofing
T	Tampering
R	Repudiation
I	Information disclosure / leakage
D	Denial of service
E	Elevation of privilege

Microsoft Threat Modelling Tool

https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool

Source: en.wikipedia.org/wiki/STRIDE_(security)

Getting Started with Data Loss Prevention

Richard Harpur
INFORMATION SECURITY PROFESSIONAL, CISM

@rharpur www.richardharpur.com

https://www.pluralsight.com/library/courses/getting-started-data-loss-prevention/table-of-contents

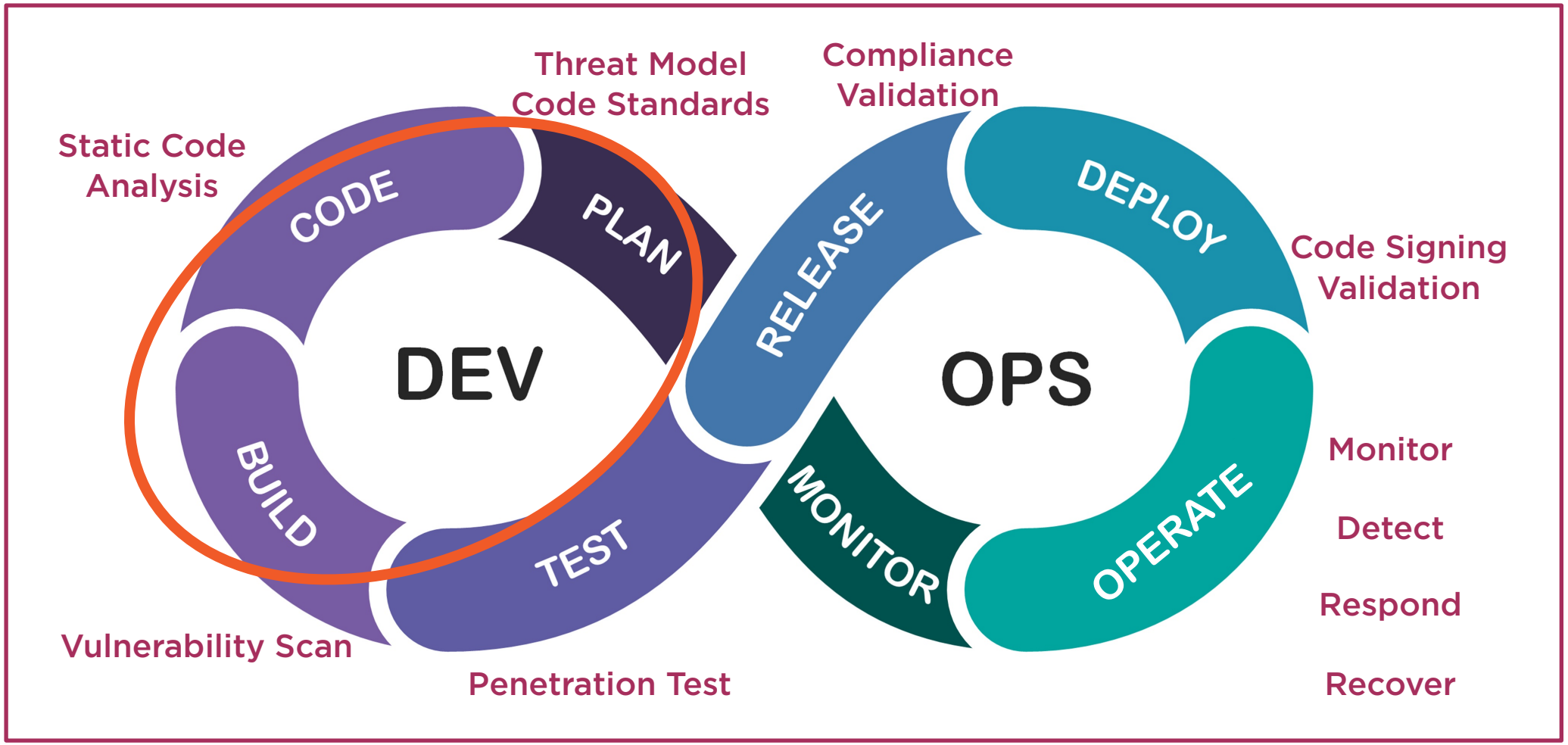
Secure Code Standards

CMU SEI -Top 10 Secure Coding Practice

1. Validate input
2. Heed compiler warnings
3. Architect and design for security
4. Keep it simple
5. Default deny
6. Adhere to principle of least privilege
7. Sanitize data from other systems
8. Practice defense in depth
9. Practice effective quality assurance
10. Adopt a secure coding standard

https://wiki.sei.cmu.edu/confluence/display/seccode/Top+10+Secure+Coding+Practices

Positioning DevSecOps in Your Lifecycle



SAST or SCA

Static Application Security Testing (SAST)

Examines source code to identify weaknesses that can lead to security vulnerabilities

Software Composition Analysis (SCA)

Checks Open Source components against known vulnerabilties

Secure Code Analysis

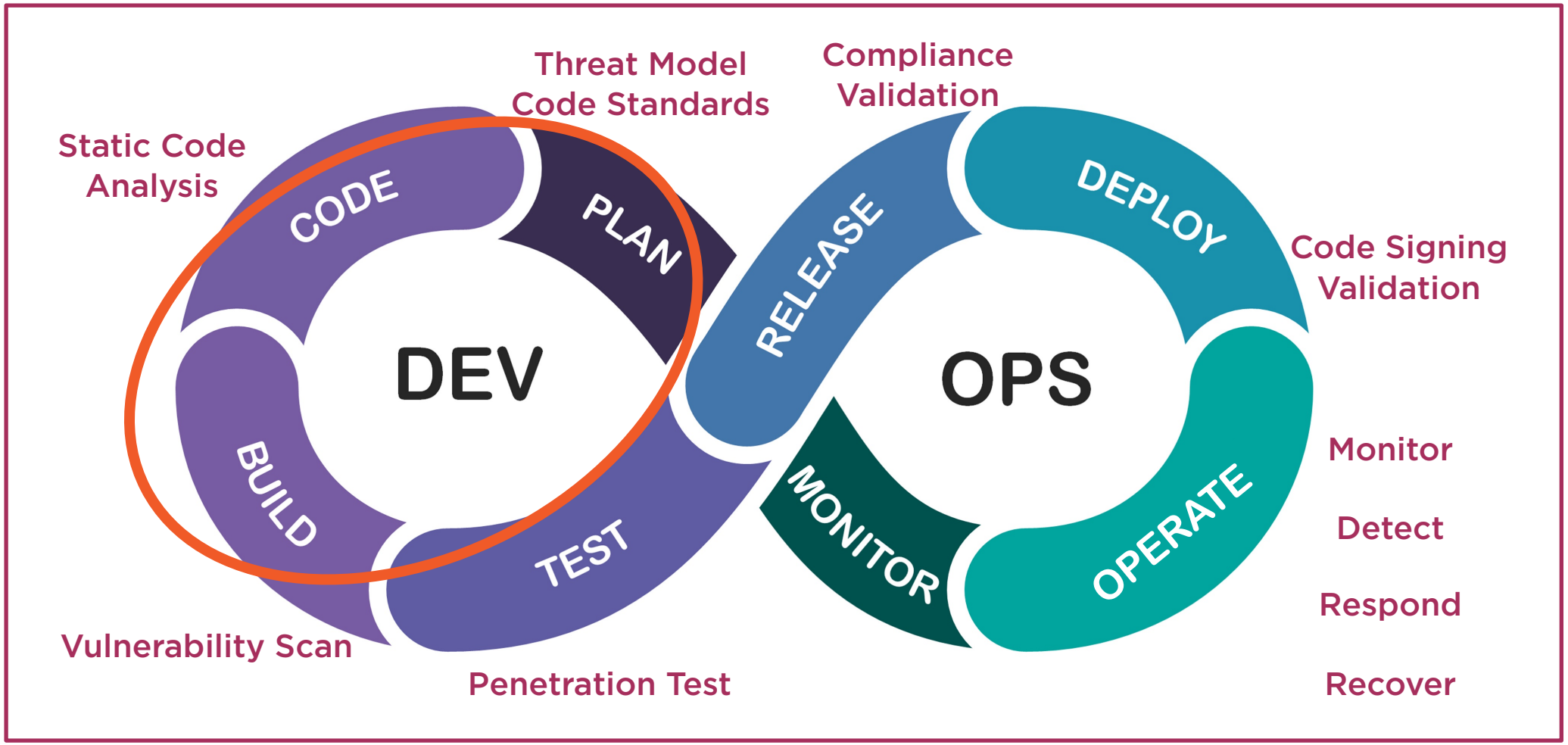
Features of SAST

1. Reads source code
2. Language specific scanner
3. False positives
4. Fast and automated
5. Finds weaknesses early

NIST list of source code security analyzers

https://samate.nist.gov/index.php/Source_Code_Security_Analyzers.html

Positioning DevSecOps in Your Lifecycle



Vulnerability Scanning

Software Composition Analysis (SCA)

Checks Open Source components against known vulnerabilities

Dynamic Application Security Testing (DAST)

Vulnerability scanners run on completed (compiled) code

Summary

Multiple checks to ensure secure code

Checks at design time

Checks at code time

Checks at code complete

Up Next

- Designing DevSecOps for Test, Release, and Operate SDLC phases