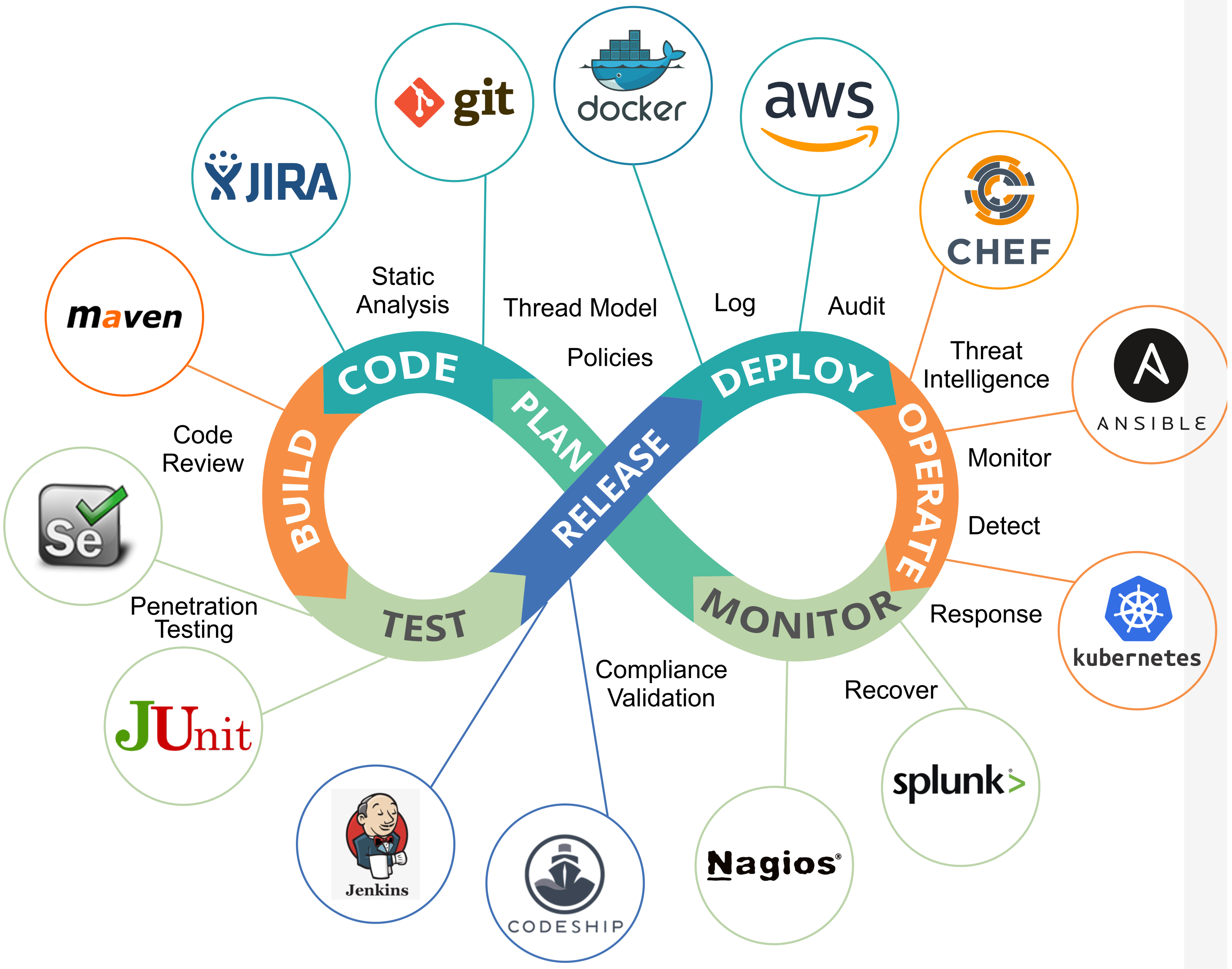
GETTING STARTED
DevSecOps Basics

Why is DevSecOps?

DevSecOps is a set of practices of adding security components to each step of the DevOps process. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality while taking care of the security aspect.



Why this topic?

DevSecOps term is a relatively new term but given the rise in security awareness as well as breaches, it has become the talk of the town. Instead of having a team for manual security testing, it is way more efficient and cost/time effective to have automated security checks and catching the issue as soon as possible. This makes DevSecOps a highly demanded and regarded skill in the industry.

- Knowledge of DevOps process

What will you learn?

This section covers different components/tools used in various phases of the DevSecOps process. The objective is to teach the user how these components are used in isolation instead of providing the whole DevSecOps pipeline. The labs are focused on learning and not on CTF style evaluation. Each lab comes equipped with a PDF manual that covers the usage of the tools with different easy to follow examples.

References:

- DevSecOps: https://www.sumologic.com/insight/devsecops-rugged-devops/

Sub-sections/topics to be covered

We are going to cover the following stages of the DevOps process:

1. Automated Code Review

The Automated Code Review (ACR) phase scans the code for known mistakes, security issues, and vulnerabilities. This is way more efficient and easily scalable than Manual Code Review done for the same purpose. It is not to be confused with Peer Code review practice that is done to improve the code quality and find business/logical/flow mistakes.

2. Sensitive Information Scan

The Sensitive Information Scan (SAS) phase scans the code for sensitive information (e.g. hardcoded password, tokens, secret keys, etc) before pushing the code into code repositories. This makes sure that even if the code falls into wrong hands tomorrow, the sensitive information won't get exposed.

3. Static Application Security Testing

The Static Application Security Testing (SAST) is done to identify the possible vulnerabilities or security issues in non-running source code by using techniques likeTaint Analysis and Data Flow Analysis.

4. Dynamic Application Security Testing

The Dynamic Application Security Analysis (DAST) is performed to identify the possible run-time vulnerabilities or security issues. Unlike static analysis, dynamic analysis is performed on a running project.

5. Software Composition Analysis

Software Composition Analysis is performed to identify the dependency packages/libraries for the project and check those against known vulnerabilities.

6. Vulnerability Management

Vulnerability Management (VM) is the process of identifying the inventory, using tools to perform security tests on the project to identify vulnerabilities, catalog the vulnerabilities, and then patching/fixing those. It is an ongoing process and can be thought of as a part of continuous security testing.

7. Compliance as Code

Compliance as Code (CAC) is an approach to automate the enforcement of (IT) compliance regulations by writing policies in a file. These policies can then be enforced or checked in an automated manner.