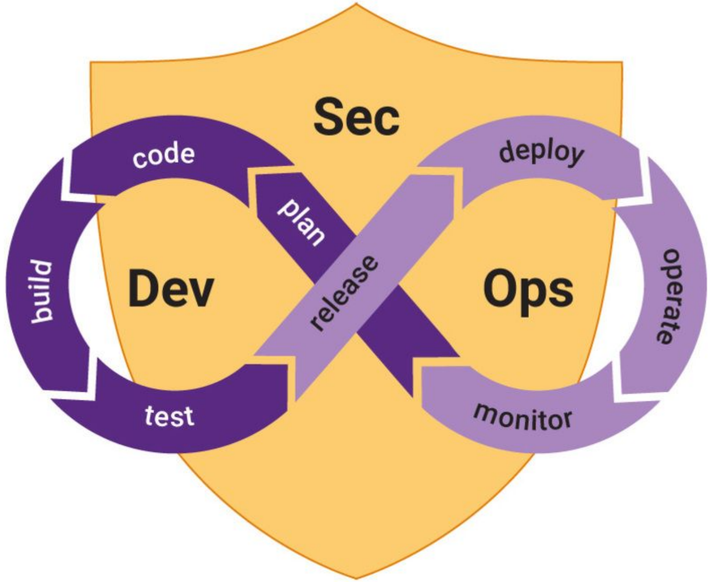
ENSA KHOURIBGA ECOLE NATIONALE DES SCIENCES APPLIQUÉES



DevSecOps * in 1

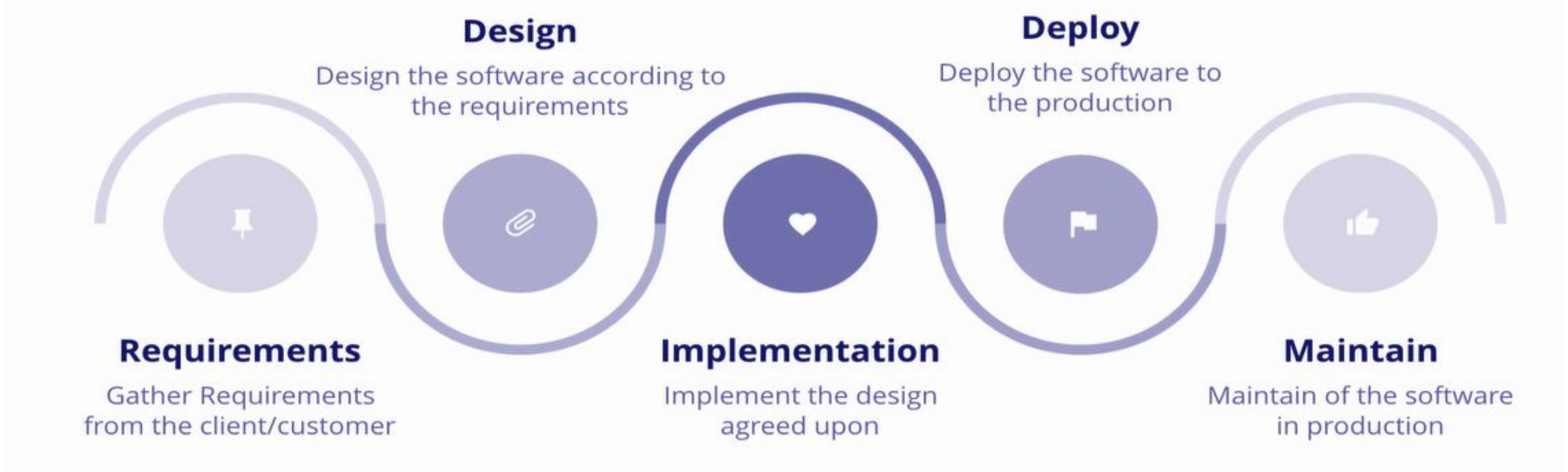
What? Why and How?

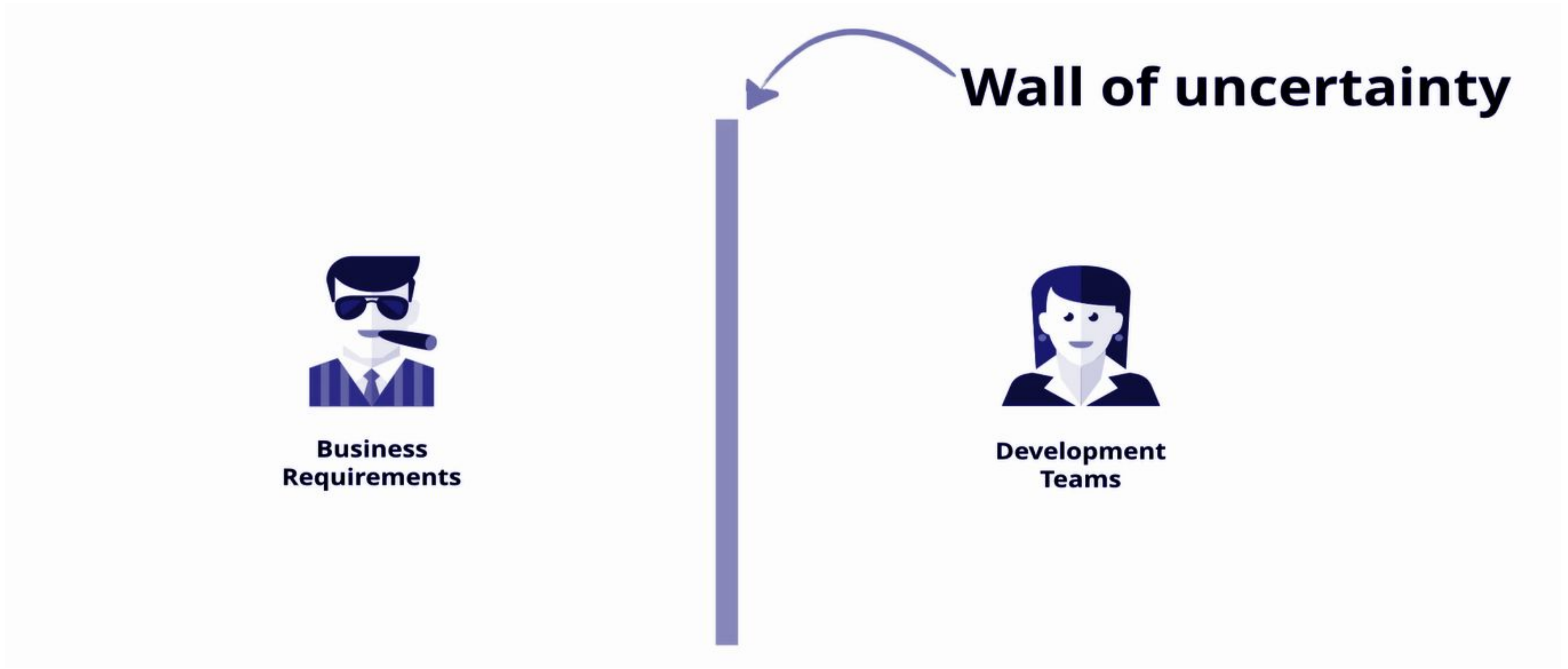
NAJIM Ayoub Offensive Security Engineer | Consultant | Certified DevSecOps Proffesional "CDP"

Plan

Introduction

Traditional SDLC


Design
Design the software according to the requirements
Deploy
Deploy the software to the production
Requirements
Gather Requirements from the client/customer
Implementation
Implement the design agreed upon
Maintain
Maintain of the software in production


Wall of uncertainty
Business Requirements
Development Teams

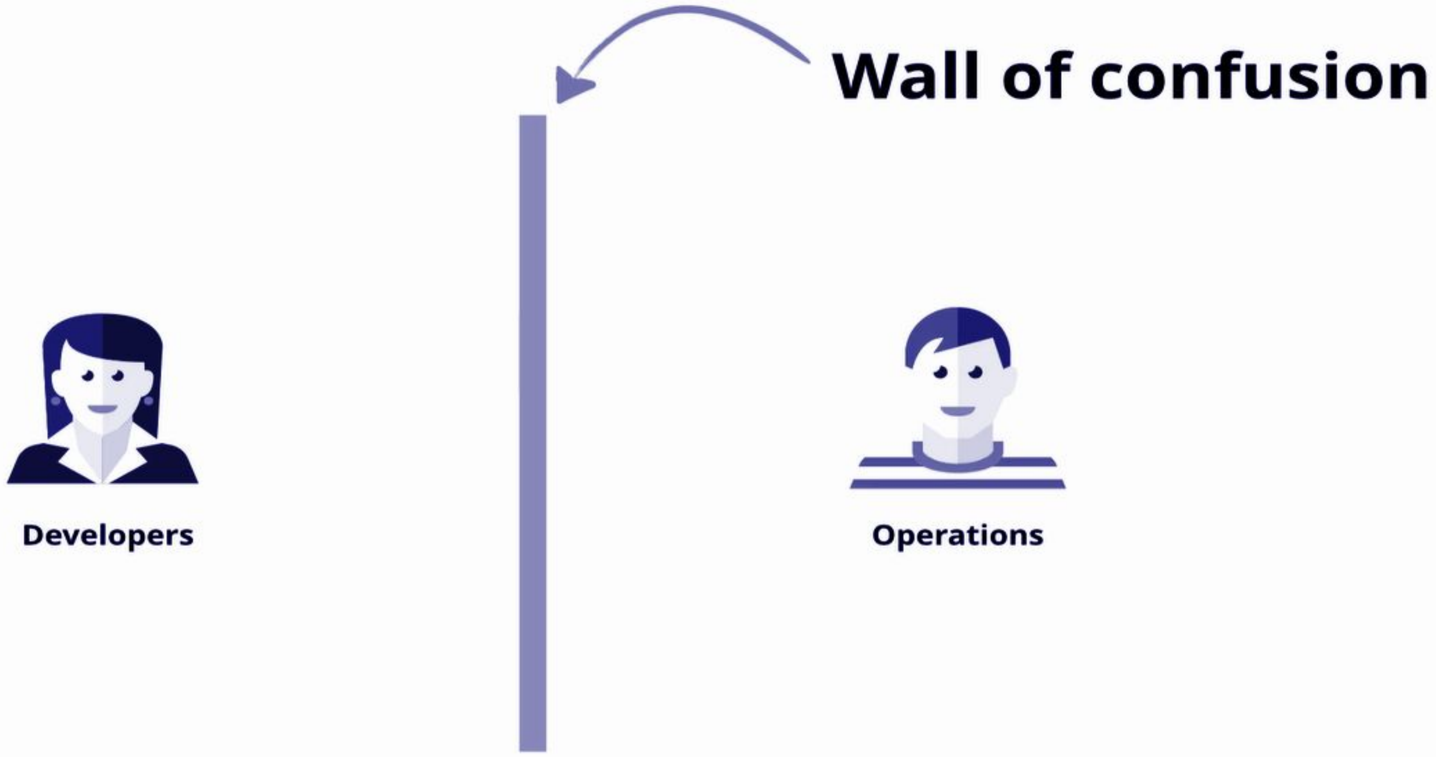
Then Agile Happened

Everything changed after agile, much shorter development cycles and faster deploys to production.

Speed with which changes are being made is beyond security's (operations) 🚨 reach.

Enter the change

Agile

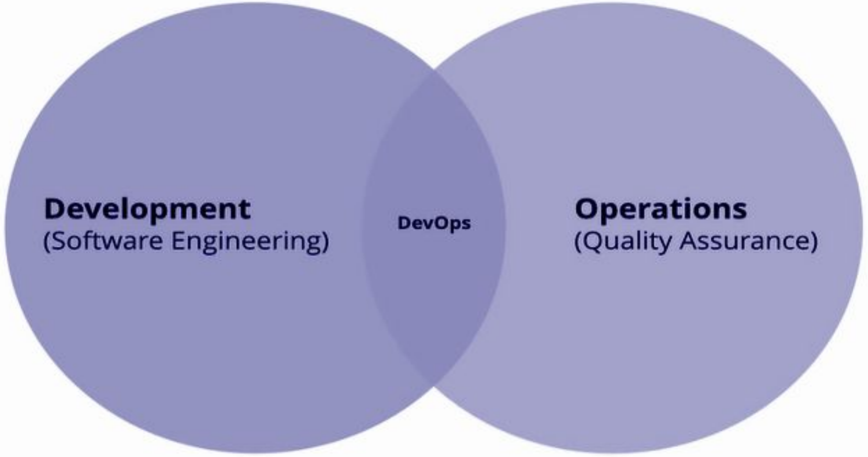

Wall of confusion
Developers
Operations

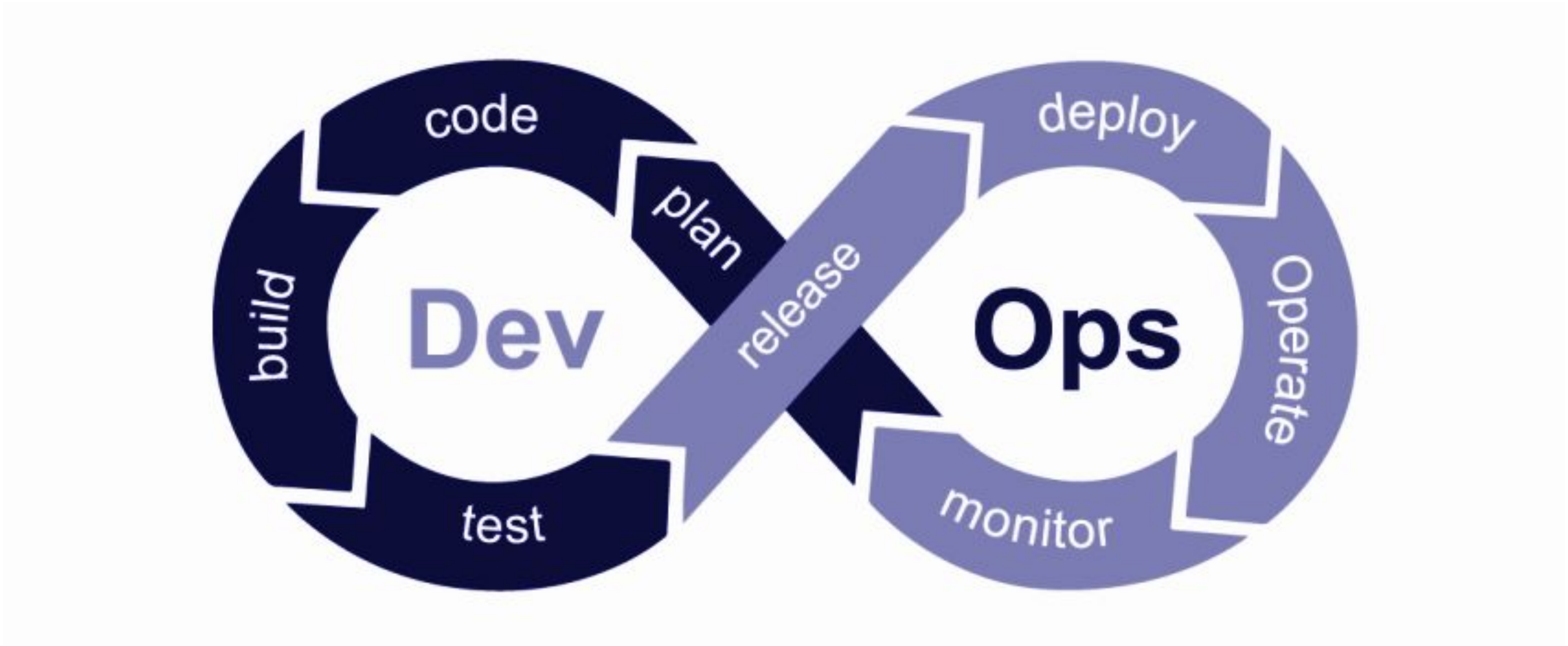
Principles & Concepts

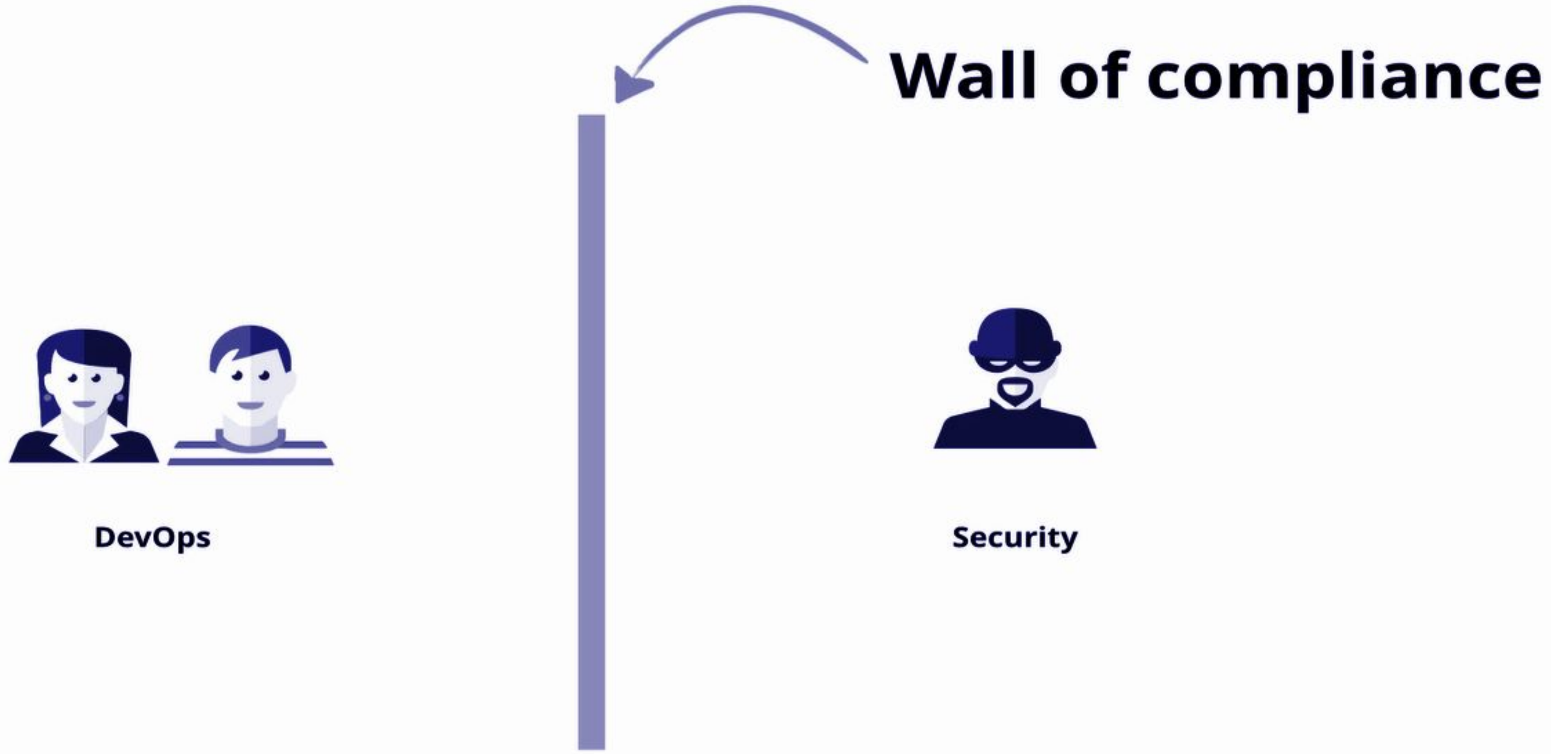
DevOps

DevOps is a software engineering practice that aims at unifying software development (Dev) and software operation (Ops). - *wikipedia*

DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality - *Bass, Weber, and Zhu*



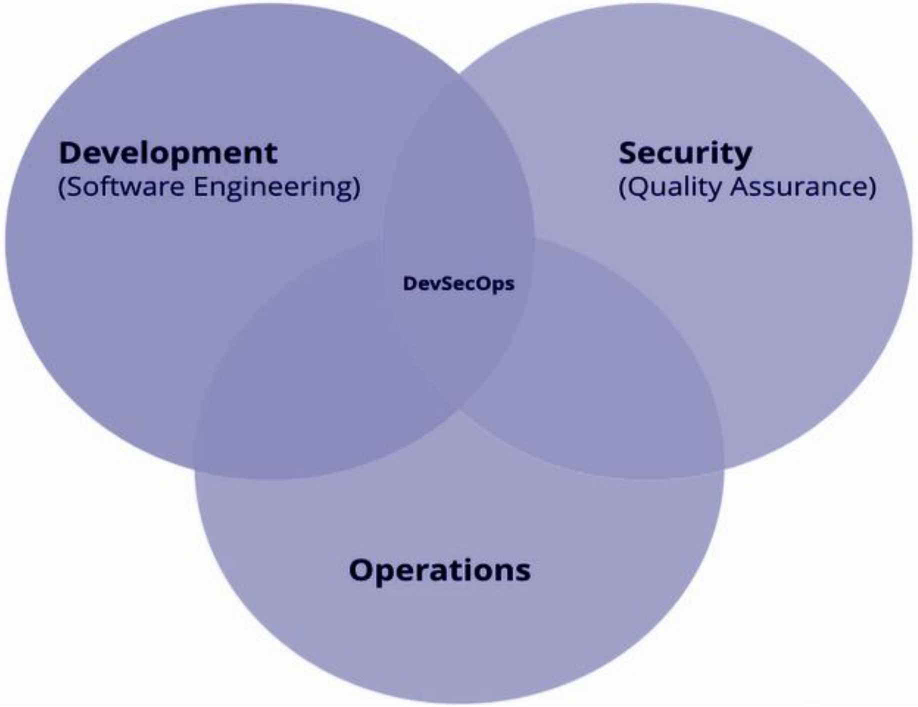

code
plan
build
Dev
test
release
deploy
Ops
Operate
monitor


Wall of compliance
DevOps
Security

DevSecOps

DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality - *Bass, Weber, and Zhu*

By definition, security is part of DevOps.



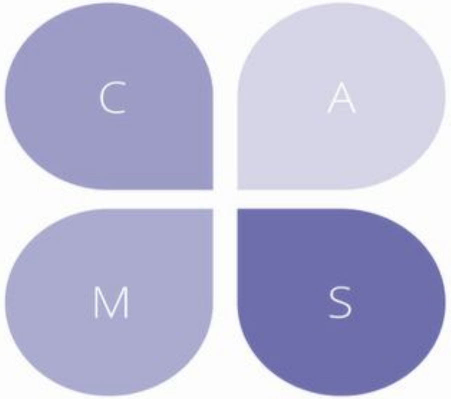
How to DevSecOps ?

Culture

DevOps is about breaking down barriers between teams; without culture other practices fail

Automation

Often mistaken as DevOps itself but a very important aspect of the initiative.



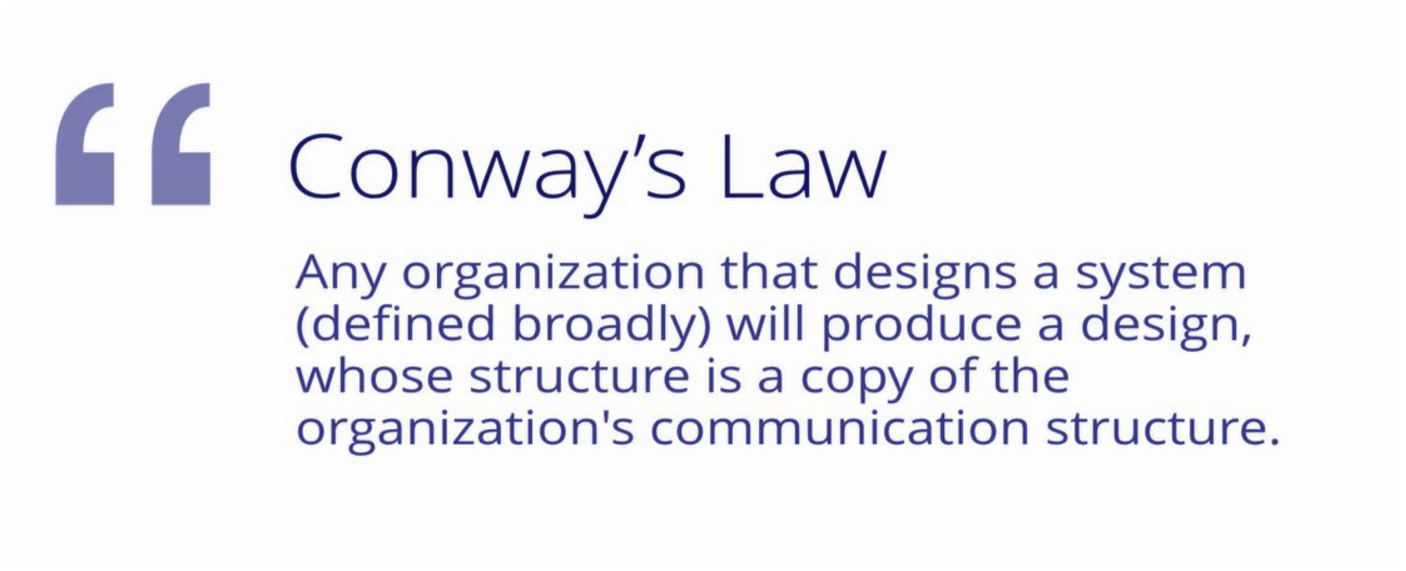
Measurement

Measuring activities in CI/CD helps in informed decision making among teams

Sharing

Sharing tools, best practices etc., among the teams/organization improves confidence for collaboration.

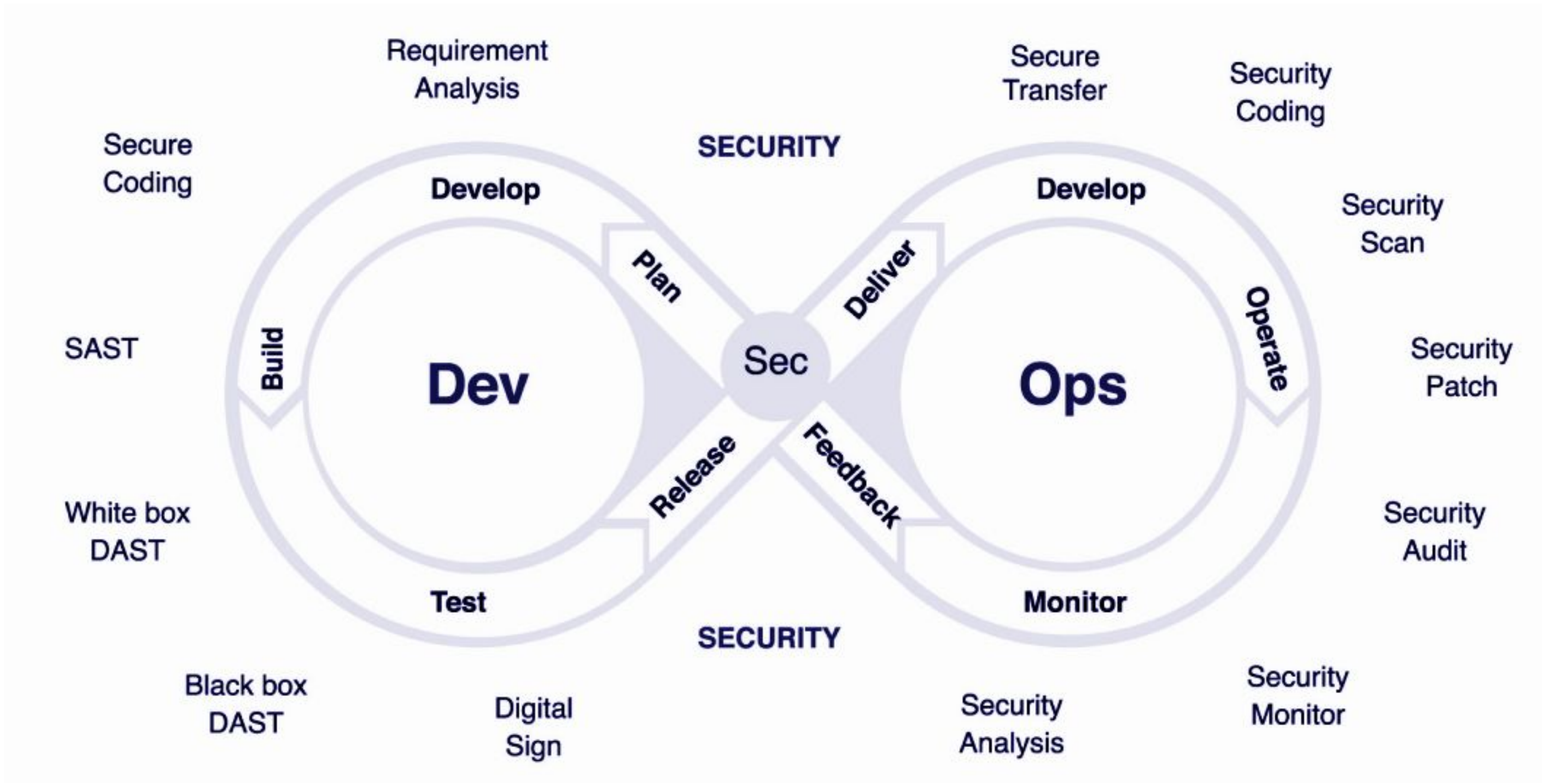
Core Values of DevOps

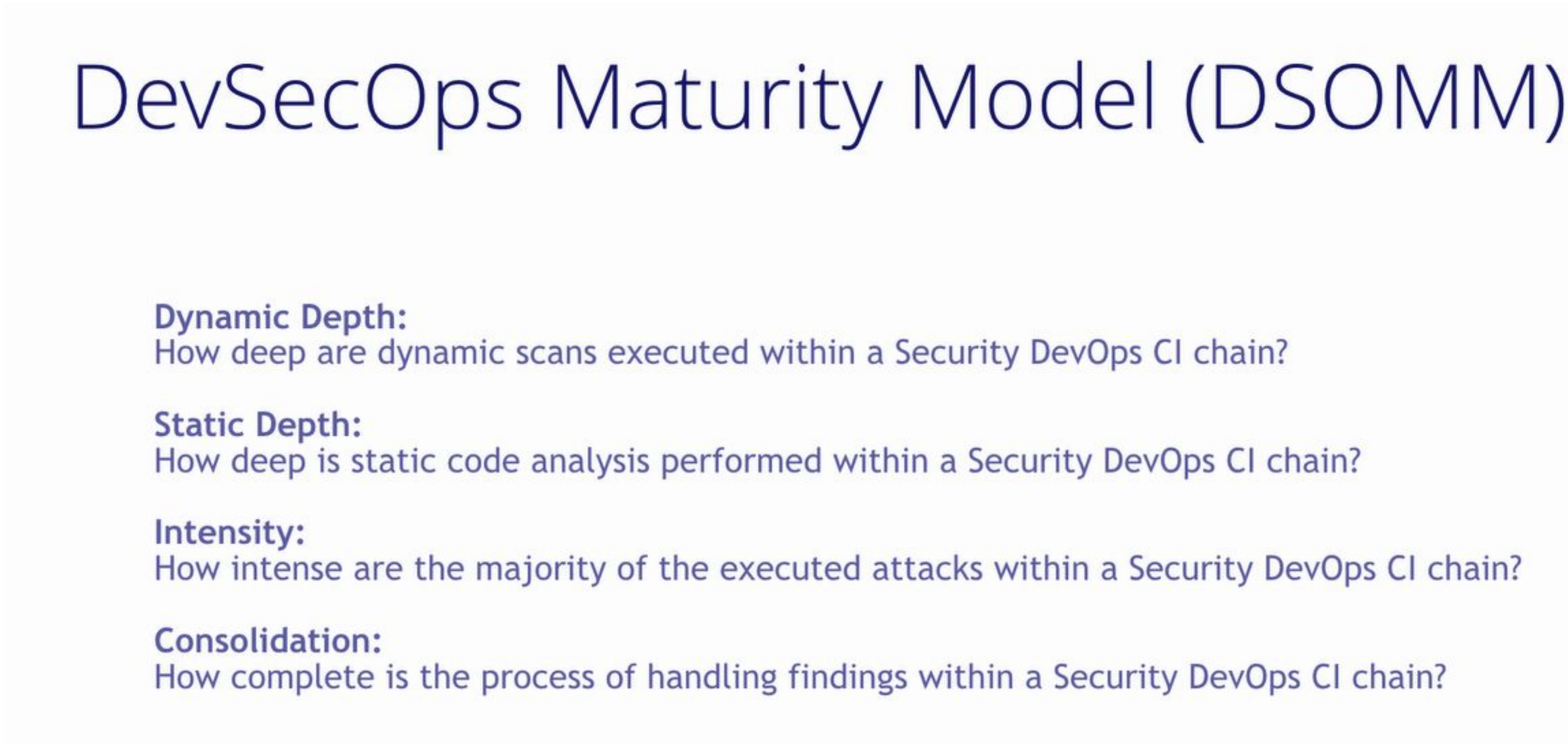


Conway’s Law

Any organization that designs a system (defined broadly) will produce a design, whose structure is a copy of the organization's communication structure.

DevSecOps Stages


Requirement Analysis
Secure Coding
SAST
White box DAST
Black box DAST
Digital Sign
SECURITY
Develop
Plan
Build
Dev
Test
Release
Sec
Deliver
Feedback
Develop
Operate
Ops
Monitor
SECURITY
Secure Transfer
Security Coding
Security Scan
Security Patch
Security Audit
Security Monitor
Security Analysis



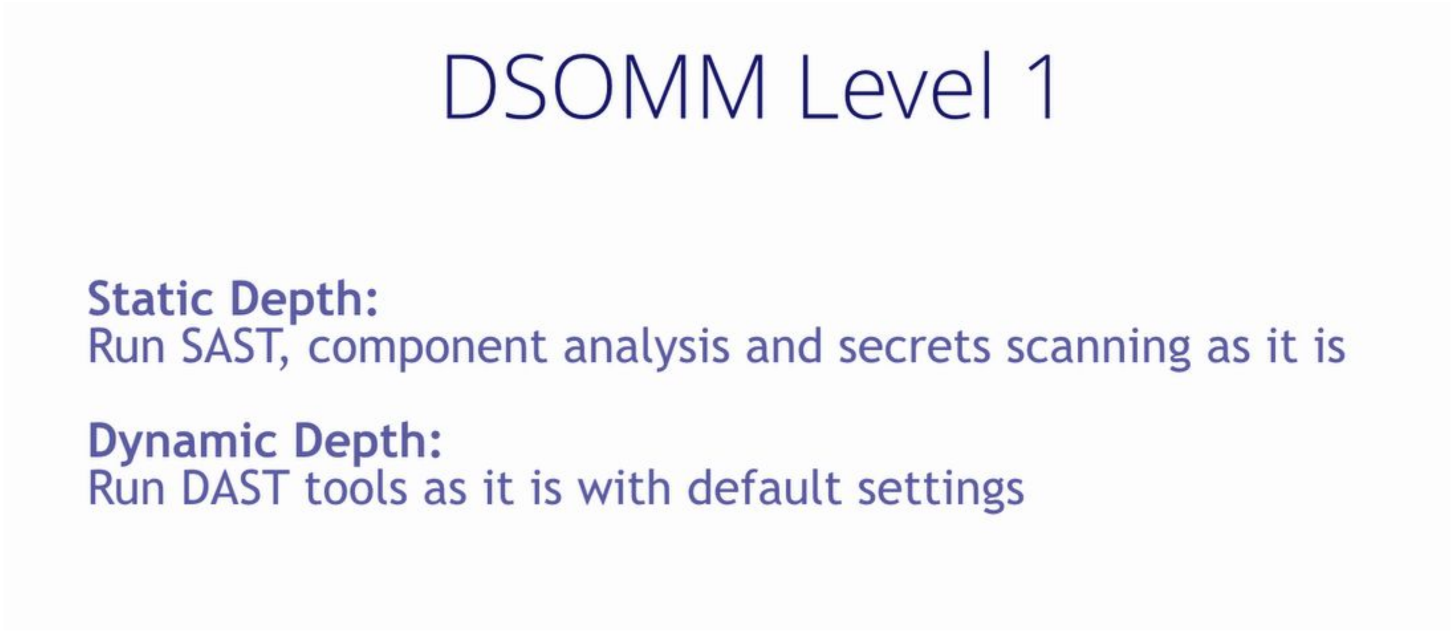
DevSecOps Maturity Model (DSOMM)

Dynamic Depth:
How deep are dynamic scans executed within a Security DevOps CI chain?

Static Depth:
How deep is static code analysis performed within a Security DevOps CI chain?

Intensity:
How intense are the majority of the executed attacks within a Security DevOps CI chain?

Consolidation:
How complete is the process of handling findings within a Security DevOps CI chain?



DSOMM Level 1

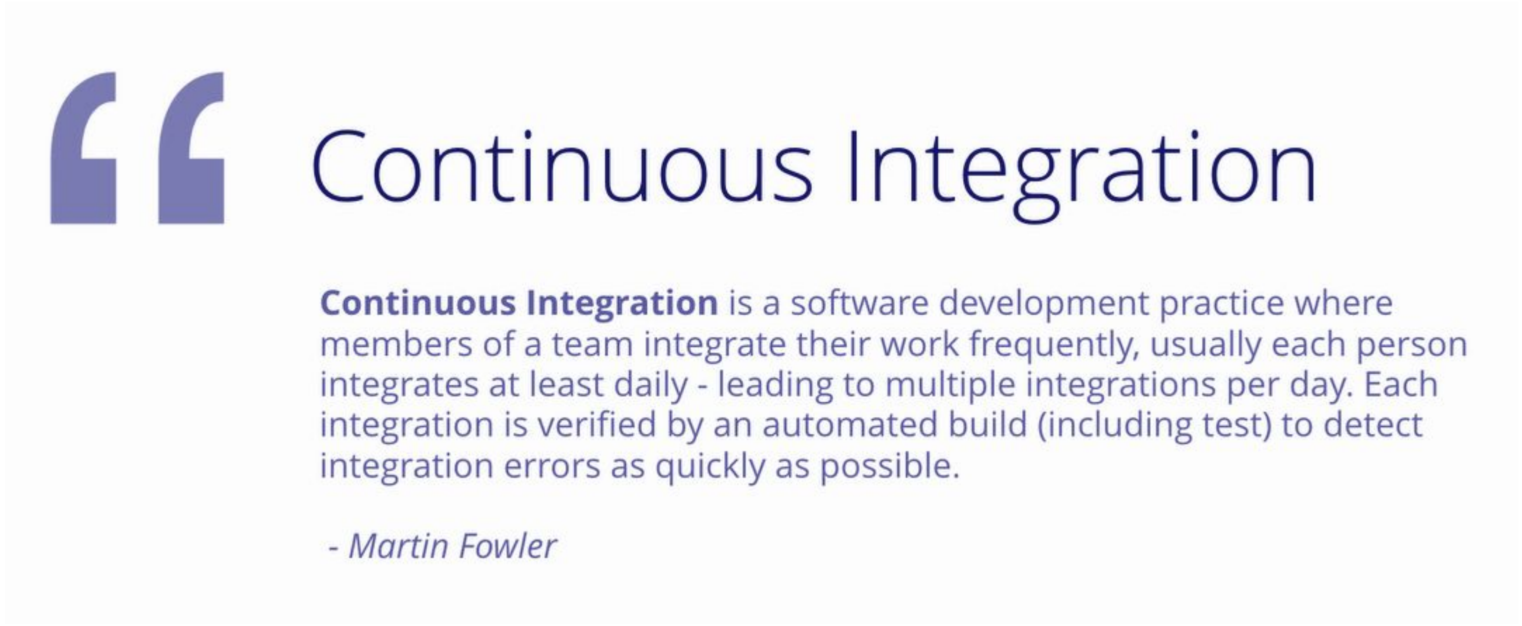
Static Depth:
Run SAST, component analysis and secrets scanning as it is

Dynamic Depth:
Run DAST tools as it is with default settings

DSOMM Level 2

Static Depth:
Run SAST, component analysis and secrets scanning with minor tweaks to the rulesets

Dynamic Depth:
Run DAST tools with minor tweaks to tools.



“ Continuous Integration

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.

- Martin Fowler

“ Continuous Delivery

Continuous Delivery is a software engineering approach in which continuous integration, automated testing, and automated deployment capabilities allow software to be developed and deployed rapidly, reliably and repeatedly with minimal human intervention. Still, the deployment to production is defined strategically and triggered manually.

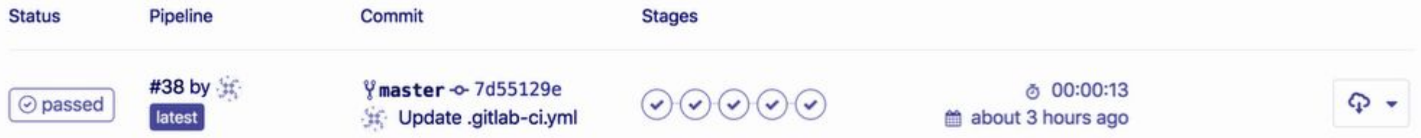
- Martin Fowler

“ Continuous Deployment

Continuous Deployment is a software development discipline where you build software in such a way that the software is released to production automatically without any human intervention. This uses Continuous Delivery pipeline but deploys automatically to production if tests pass.

Gitlab CI/CD Pipelines

1. A pipeline is a system consisting of one or more stages to continuously integrate/delivery/deploy software.

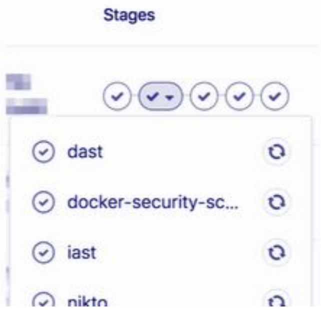


Status	Pipeline	Commit	Stages		
✓ passed	#38 by latest	master -o- 7d55129e Update .gitlab-ci.yml	✓ ✓ ✓ ✓ ✓	00:00:13 about 3 hours ago	

2. A stage (build, test, deploy) is a combination of jobs to achieve goal of a stage.

3. Jobs in a stage are run in parallel and on success, the pipeline moves on to the next stage.

If one of the jobs fails, the next stage is not (usually) executed.



Create your first CI/CD pipeline

Add **.gitlab-ci.yml** for a project

$ cat .gitlab-ci.yml

```
stages:
  - build
  - test
  - integrate
  - deploy

Job1:
  stage: build
  script:
    - echo "This is a build step"
Job2:
  stage: test
  script:
    - echo "This is a test step"
    - exit 1

job3:
  stage: integrate
  script:
    - echo "This is a integrate step"
```

We can do this from command line

```
$ git add .gitlab-ci.yml
$ git commit -m "Add .gitlab-ci.yml"
$ git push origin master
```

Steps: https://github.com/teacheraio/DevSecOps-Studio/wiki/Lesson-two:-Setting-up-CI-CD-pipeline

See Google Document

Demo - SCA, SAST, DAST and VM

Thank you!!