OWASP Indonesia Meetup

Strengthen and Scale security using DevSecOps

www.teachera.io @secfigo secfigo@gmail.com

Mohammed A. **Imran**

Senior Security Engineer

whoami

Author, Speaker and Community Leader.

Speaker/Trainer at Blackhat, AppSec EU, Pycon, All Day DevOps, DevSecCon London, DevSecCon Singapore, Nullcon etc.,

Organizer of DevSecOps Track in OSS 2018.

Project Leader for OWASP DevSecOps Studio, DevSlop, Integra and Awesome-Fuzzing projects.

Organised around 100 monthly security meetings and about 50 workshops.

SCJP, OSCP, OSCE. AWS-CP, AWS-CSA, AWS-SS

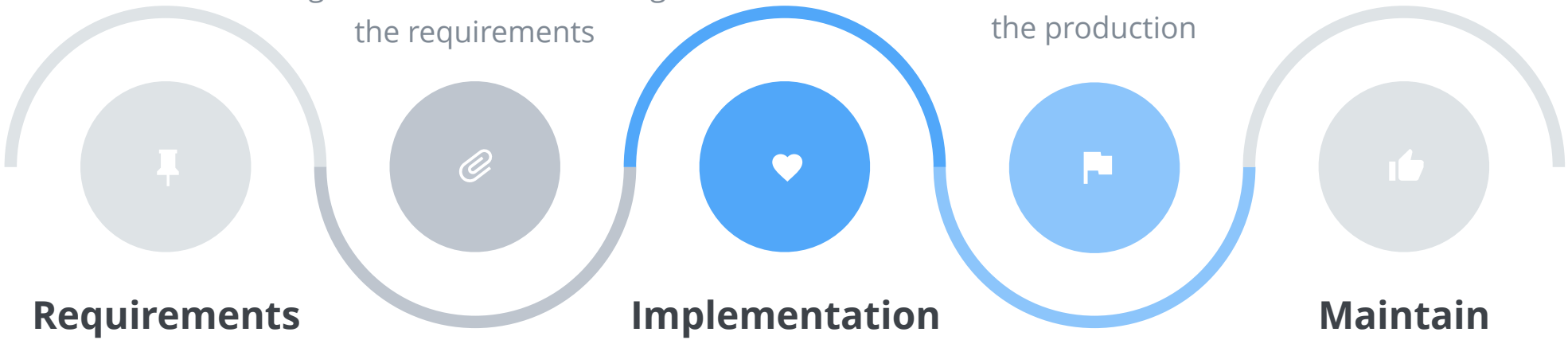
Agile and DevOps

1

Long Long time ago

Trivia: how is this related to Singapore ?

Traditional SDLC



Requirements

Gather Requirements from the client/customer

Design

Design the software according to the requirements

Implementation

Implement the design agreed upon

Deploy

Deploy the software to the production

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.

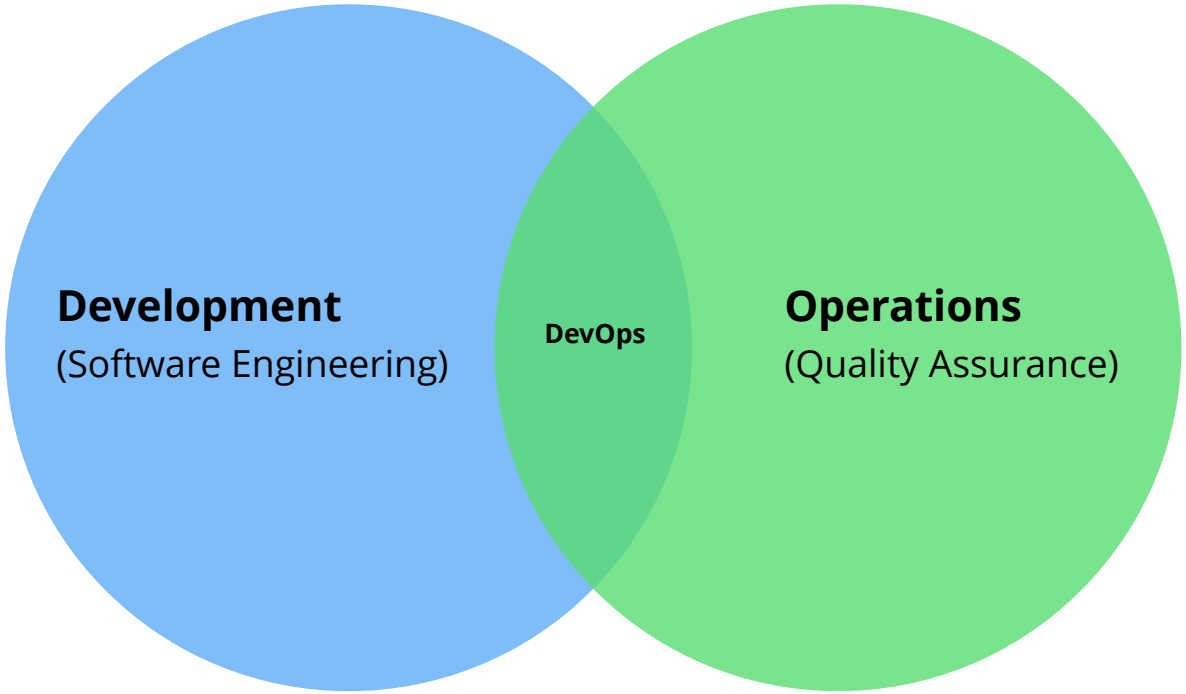
Wall of confusion

Developers

Operations

DevOps

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*



Release

D

Release the artefact as production ready after change/release approvals

Configure

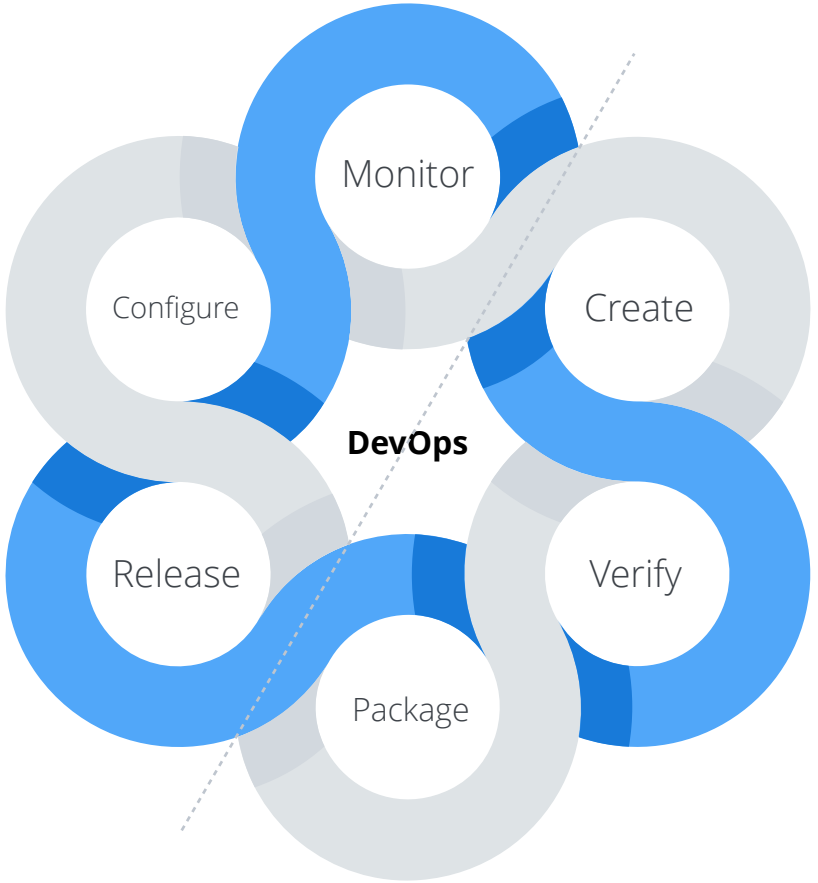
E

Configure the application/ stack using configuration management

Monitor

F

Monitor the application for its performance, security and compliance



DevOps Cycle

Plan & Create

A

Plan and implement the code using source code management (SCM)

Verify

B

Test and verify the code does, what business wants.

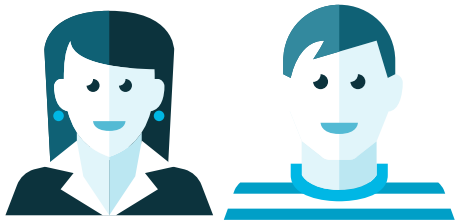
Package

C

Package the code in a deployable artifact & test it in staging environment


BEST
Do you speak Chinese

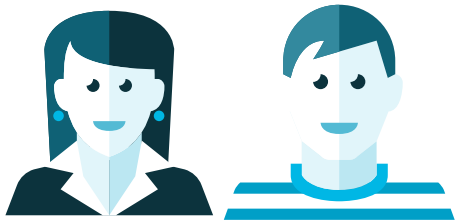
Wall of compliance



DevOps

Security

Wall of compliance



DevOps

Security

Traditional Secure SDLC

1. TRAINING	!. REQUIREMENTS	3. DESIGN	. IMPLEMENTATION	5. VERIFICATION	6. RELEASE	7. RESPONSE
1. Core Security Training	2. Establish Security Requirements	5. Establish Design Requirements	8. Use Approved Tools	11. Perform Dynamic Analysis	14. Create an Incident Response Plan	Execute Incident Response Plan
	3. Create Quality Gates/Bug Bars	6. Perform Attack Surface Analysis/ Reduction	9. Deprecate Unsafe Functions	12. Perform Fuzz Testing	15. Conduct Final Security Review	
	4. Perform Security and Privacy Risk Assessments	7. Use Threat Modeling	10. Perform Static Analysis	13. Conduct Attack Surface Review	16. Certify Release and Archive	

Security is Outnumbered!

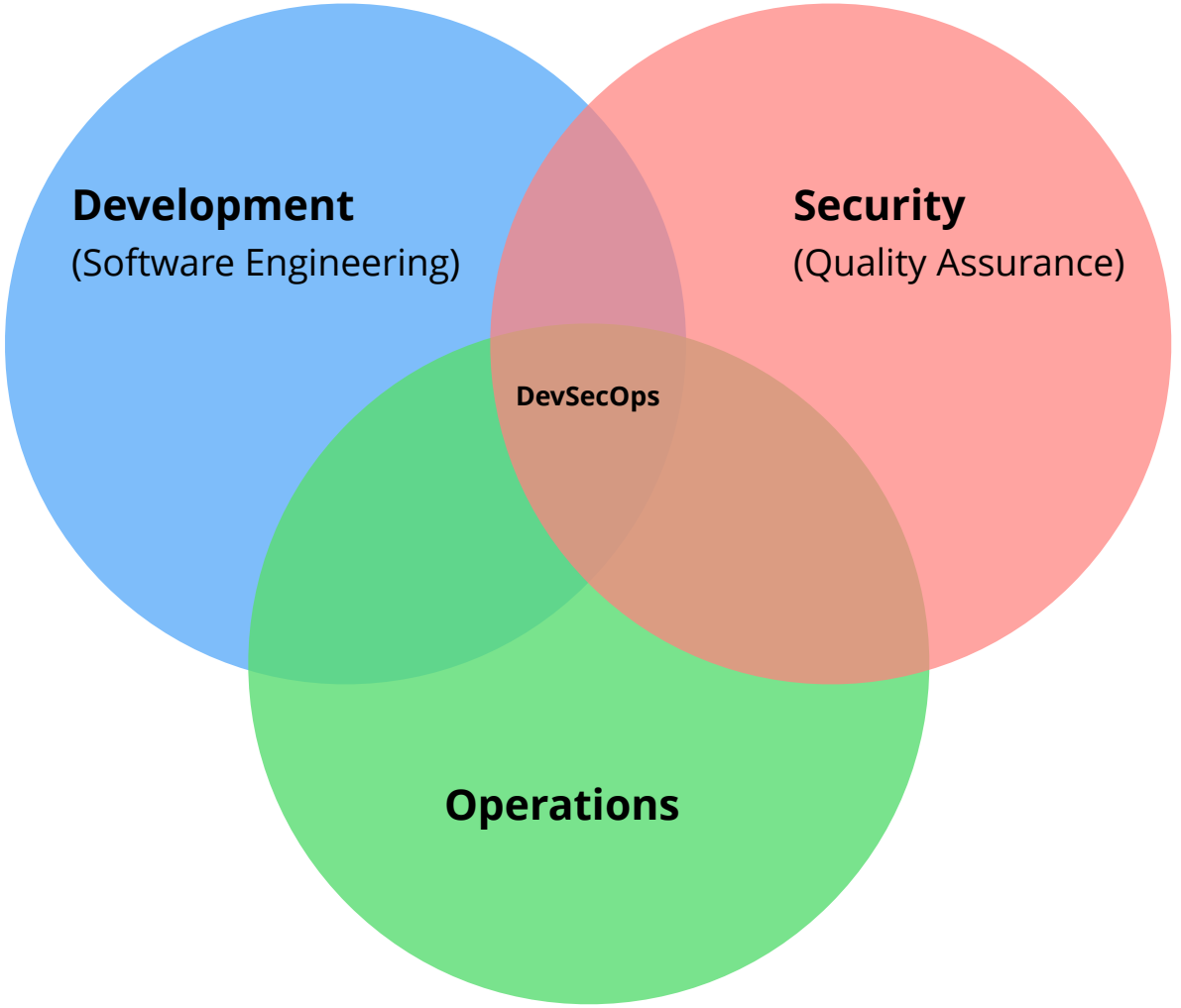
Dev / Ops / Security

100 / 10 / 1

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.



Resilience

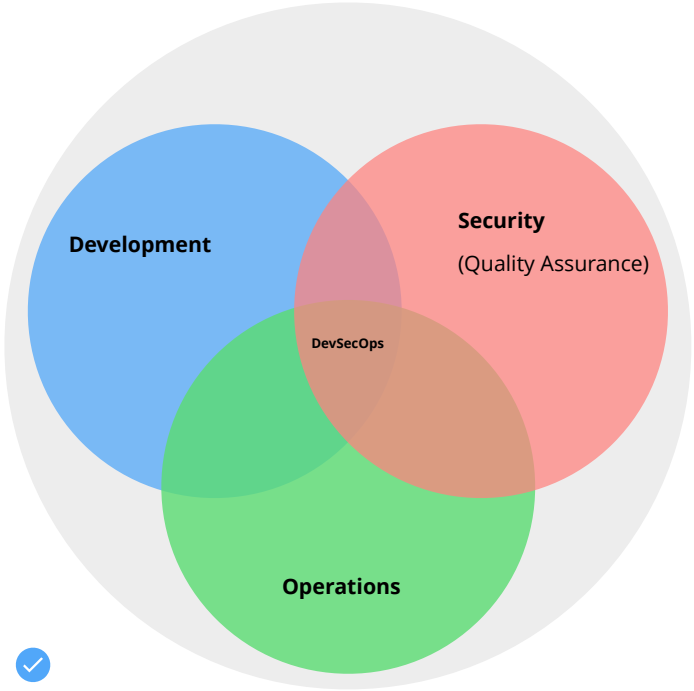
DevOps helps organisations in designing and implementing resilient systems.

Speed

Speed is **competitive advantage** and DevOps helps to go to market faster.

Automation

Automation helps to reduce complexity of modern systems and can **scale** as per needs



DevSecOps Benefits

Flexibility

With ever changing technology, businesses have to be flexible and fast to deliver value to their customers otherwise **they risk** losing the **business.**

Reliability

Customers need more reliable & available systems. DevOps reduces failure rates and provides faster **feedback**

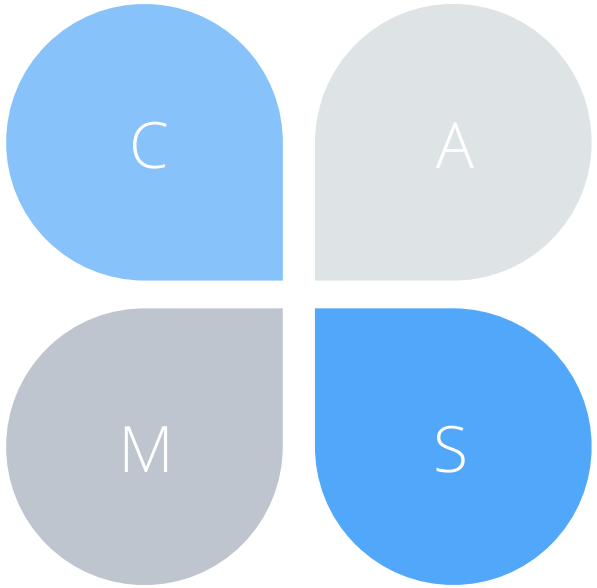
How to DevSecOps ?

Culture

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

Measurement

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



Core Values of DevOps

Automation

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

Sharing

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

Build bridges, not walls!

Build guard rails, not gates!

Embed security early and often

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.

Continuous Integration/Deployment 2

CI/CD

PLAN

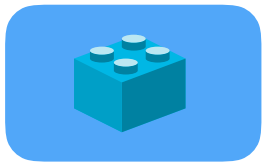
Requirements

Functional req.
Non Functional req.
Design

CODE

Code Repository

Code
Branching
Third party components
Hooks



BUILD

CI Server

Compile
Basic tests
Lint(analyze)
Package
Security

TEST

Integration Testing

Integration
Performance
Security

RELEASE

Artefact Repository

Test on staging
Release
Schedule

DEPLOY

CD Orchestration

Configuration
Inventory
Infrastructure

OPERATE

Monitor

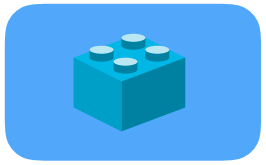
Metrics
Monitoring
Alerting

PLAN

Requirements

CODE

Code Repository



BUILD

CI Server

TEST

Integration Testing

RELEASE

Artefact Repository

DEPLOY

CD Orchestration

OPERATE

Monitor

Agile Development

Continuous Integration

Continuous Delivery

Continuous Deployment

DevOps/DevSecOps

Scale security with DevOps 3

DevSecOps Implementation

So far we have looked at Principles and Ideas behind DevSecOps but how do we start implementing DevSecOps ?

We can use the techniques (see towards your right hand side) discussed in this course to implement a full blown security pipeline.

Shift Security Left

Use CI/CD pipeline to embed security

Everything as Code(EAC

Compliance as Code and hardening via configuration management systems

Self Service

Gives developers and operations visibility into security activities

Secure by Default

Use secure by default frameworks and services

Security Champions

Encourage security champions to pick security tasks.

Use maturity models

Use DevSecOps Maturity Models to improve further

1. Shift Security left

Use CI/CD pipeline to embed security early on

DevOps: Typical Activities

PLAN

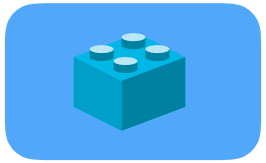
Requirements

Functional req.
Non Functional req.
Design

CODE

Code Repository

Code
Branching
Third party components
Hooks



BUILD

CI Server

Compile
Basic tests
Lint(Analyze)
Package
Security

TEST

Integration Testing

Integration
Performance
Security

RELEASE

Artefact Repository

Test on staging
Release
Schedule

DEPLOY

CD Orchestration

Configuration
Inventory
Infrastructure

OPERATE

Monitor

Metrics
Monitoring
Alerting

DevOps: Typical Security Activities

PLAN

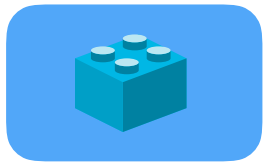
Requirements

Threat Modelling
ASVS

CODE

Code Repository

Git secrets
Dependency Scanning



BUILD

CI Server

Dependency Scanning
Code Analysis(SAST)
Security Unit Tests
Docker security Testing
Git secrets scanning
Component scanning

TEST

Integration Testing

ZAP testing - baseline
Container Scanning
Modsecurity CRS

RELEASE

Artefact Repository

Docker/Third Party
SSL scanning
Nikto/dirbuster
WPScan/JoomScan
ZAP + selenium + python
Component scanning

DEPLOY

CD Orchestration

Docker Benchmark
System Hardening
Application Hardening

OPERATE

Monitoring

Compliance as code
SOC with ELK
Verify Controls

2. Self Service

Gives developers and operations visibility into security activities

3. Security as Code (EaC)

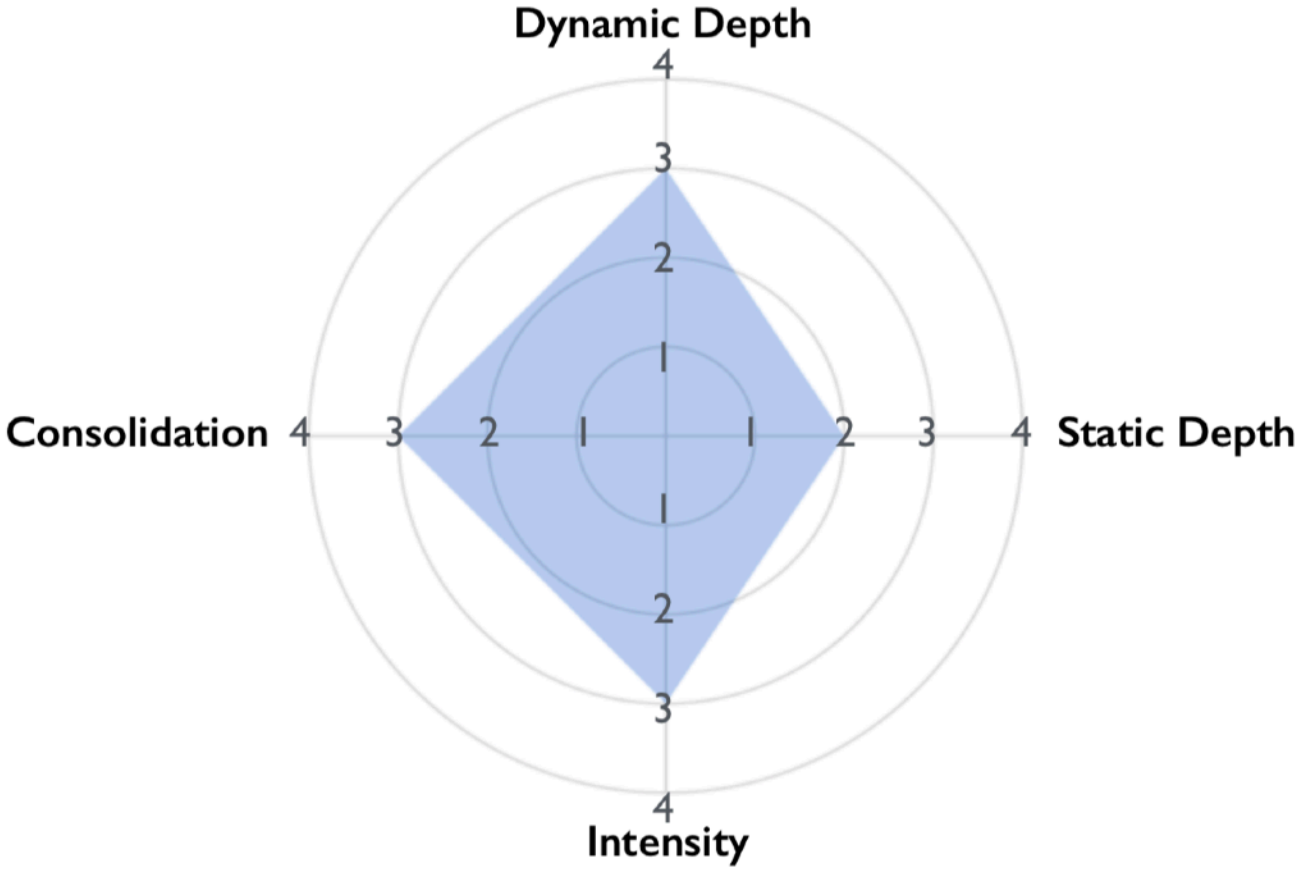
Compliance as Code and hardening via configuration management systems

4. Secure by default

Use secure by default frameworks and services

DevSecOps Maturity Model 4

DevSecOps Maturity Model (DSOMM)



Source: https://www.slideshare.net/cschneider4711/hackpra-2015-security-devops-free-pentesters-time-to-focus-on-highhanging-fruits

DevSecOps Maturity Model (DSOMM)

Static Depth: How deep is static code analysis ?

Dynamic Depth: How deep are dynamic scans executed ?

Intensity: How intense are the majority of the executed attacks ?

Consolidation: How complete is the process of handling findings ?

Source: https://www.slideshare.net/cschneider4711/hackpra-2015-security-devops-free-pentesters-time-to-focus-on-highhanging-fruits

Security Tools in CI/CD

1. Anything which takes more than 10 minutes (me being optimistic), isn't fit for CI/CD
2. SAST/DAST without creating custom rules/tweaks is of not huge benefit down the line.
3. Create separate jobs for easy debugging later.
4. Roll out tools in phases.
5. Fail builds when critical/high severity issues are found (after you have given devs/ops enough time to learn and get used to the security tools)
6. Link wiki in the scan outputs if someone needs some answers.
7. Tools which provide APIs are huge wins but make sure you at least have a CLI
8. See if your tools does incremental/baseline scans.
9. Some Ability to control the scope and false positives locally is nice (see brakeman/zap/dependency checker).
10. When in doubt ask Developers/QA for the help.
11. Everything as Code (EaC). Auditable, measurable and secure

Let's see **DevSecOps pipeline** in Action

OWASP DevSecOps Studio

DevSecOps Studio is a virtual environment to learn and teach DevSecOps concepts. Its easy to get started and is mostly automatic.

It takes lots of efforts to setup a DevSecOps environment for training/demos and more often, its error prone when done manually.

https://github.com/teacheraio/DevSecOps-Studio/

DevSecOps Studio Benefits

A **Easy to setup**

Takes only few mins to setup and start using with just one command

B **Reproducible**

The aim of this project is to setup reproducible DevSecOps Lab environment for learning and testing different tools.

C **Free & Open Source Software**

This project is a free and open software to help more people learn about DevSecOps

Our Setup for On-Premise

Push Code to git repo

Triggers Build

Run tests

Deploys to Production

Our Setup for On-Premise

Push Code to git repo

Triggers Build

Run tests

Deploys to Production

Python security tools

Security Test	Tool
SAST	Bandit
DAST	ZAP Baseline
Hardening	Ansible
Compliance	Inspec
Git Secrets	Trufflehog

Conclusion

In conclusion, we don't need large sums of money to implement DevSecOps. We can use free and open source tools to showcase the benefits and value DevSecOps provides to the organization(s).

Go on, embed security as part of CI/CD

Shift Security Left

Use CI/CD pipeline to embed security early on

Everything as Code(EAC

Use Configuration management (IaC) to implement Security as Code

Self Service

Give developers and operations visibility into security activities/tools

Secure by Default

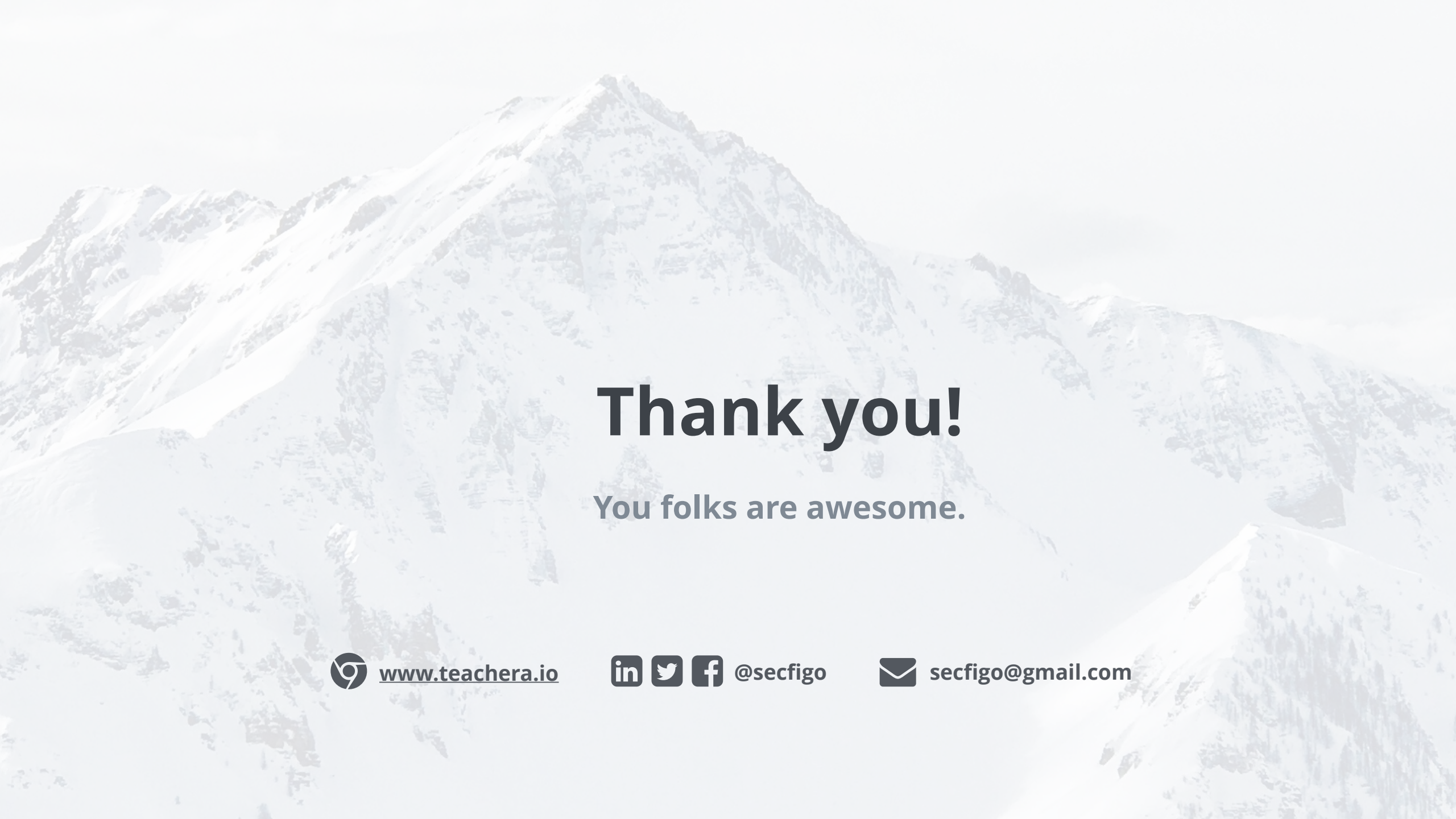
Use secure by default frameworks and services

Security Champions

Encourage security champions to pick security tasks.

Use maturity models

Use DevSecOps Maturity Models to improve further



Thank you!

You folks are awesome.

www.teachera.io @secfigo secfigo@gmail.com