

DevSecOps Fundamentos, herramientas y Automatización

Jorge García

izertis

**YOUR FUTURE,
OUR CHALLENGE.**

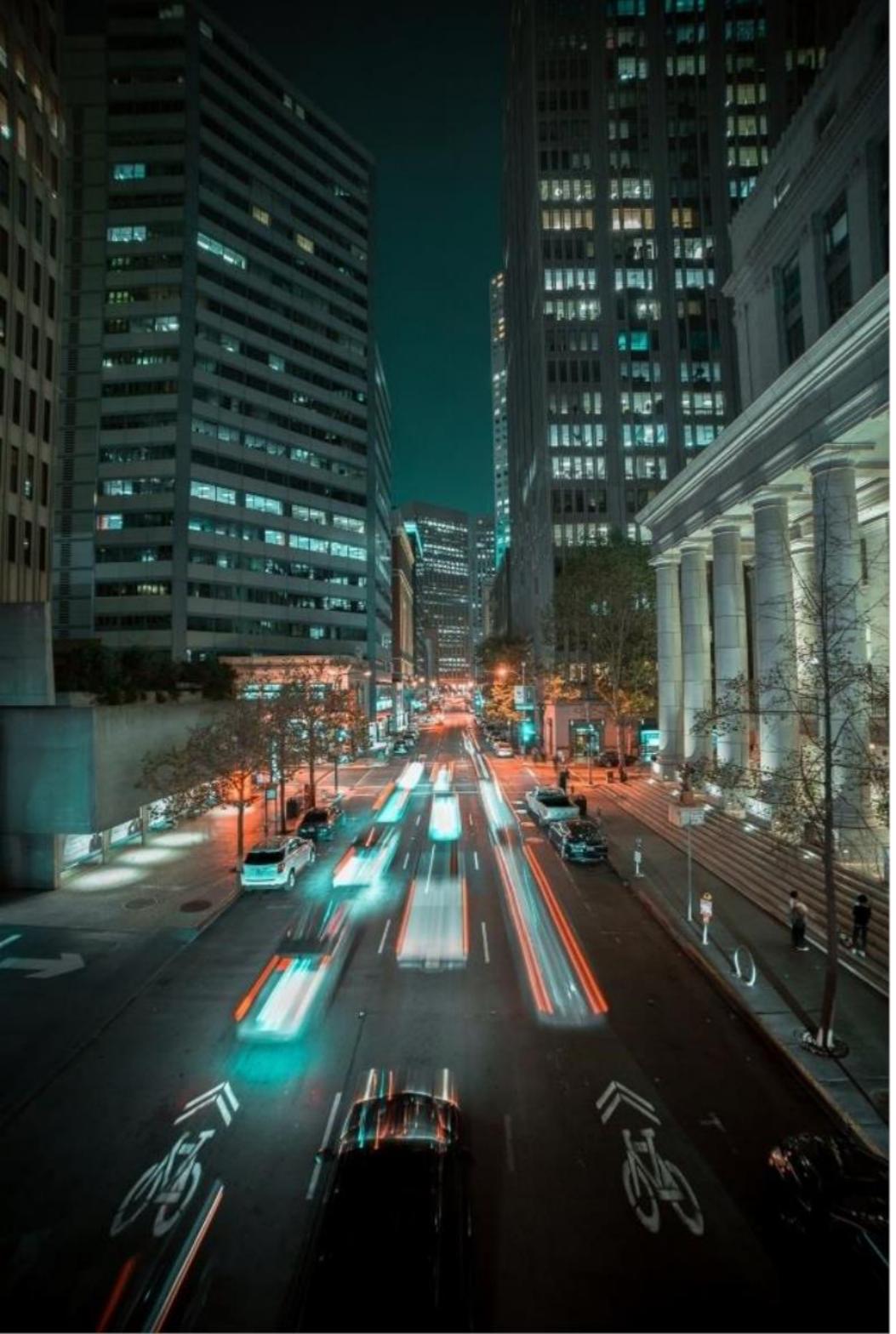


A woman with blonde hair, seen from behind, wears a large, metallic, cylindrical device on her back, possibly a life support system or a communication unit. She is standing in a field of tall grass under a dark, star-filled sky. In the foreground, a dragonfly is perched on a blade of grass, its wings catching some light. The overall atmosphere is one of a futuristic or science-fiction setting.

DIGITAL METAMORPHOSIS

BECOME THE CHANGE

SOBRE NOSOTROS



¿Quiénes somos?



Jorge García

Architect and Platform Engineer en Sidertia by Izertis.

- Más de 25 años de experiencia en el desarrollo de software en distintos ámbitos: desde el IOT hasta arquitecturas basadas en micro-servicios
- Aficionado a la electrónica, la lectura, los comics, los videojuegos, pero, sobre todo, al aprendizaje

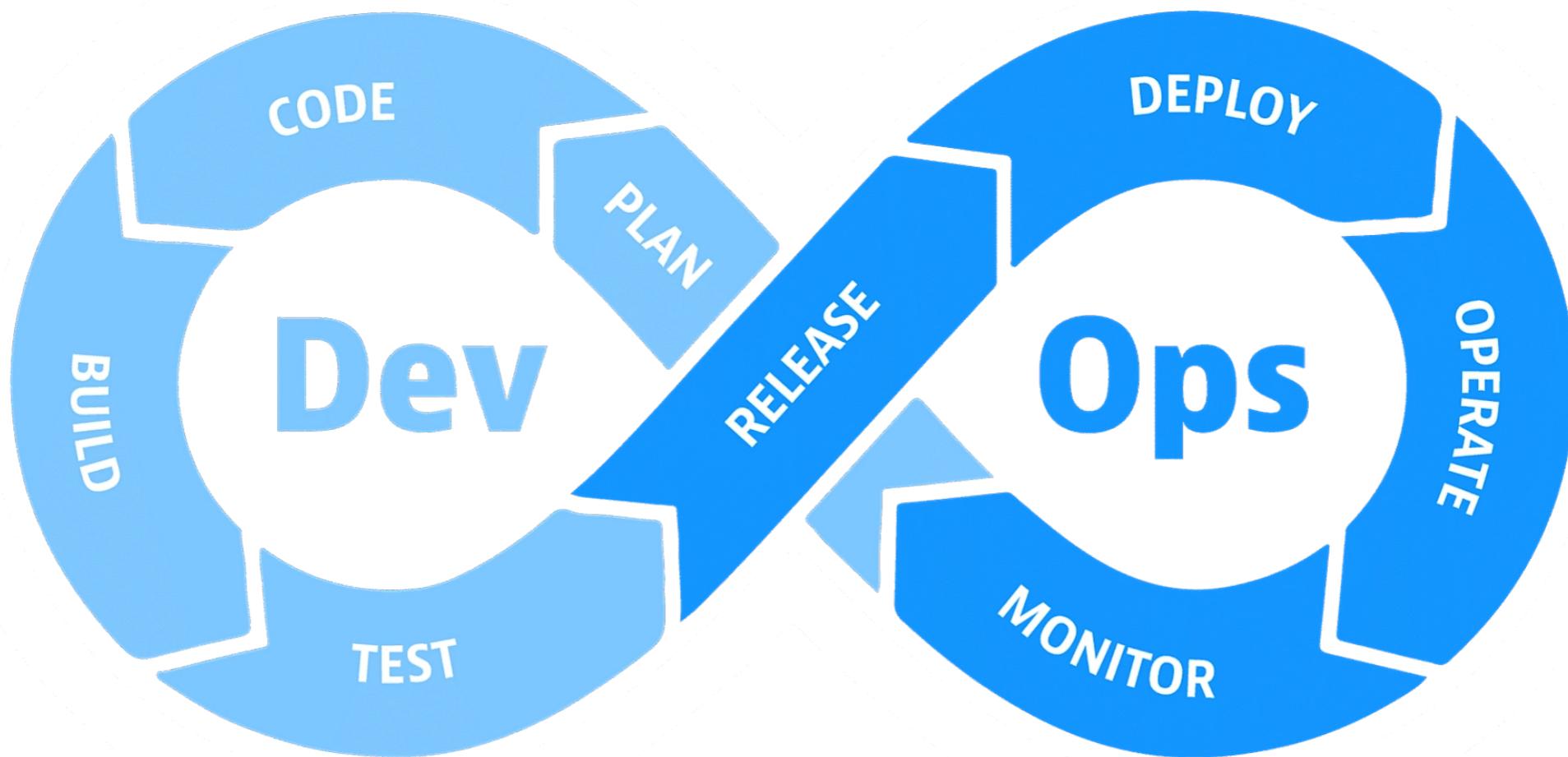
Recursos

- Conexión a Internet
 - Git client
 - Docker Desktop
 - VsCode
 - Azure free account
-
- Proyecto:
 - <https://github.com/jorgegciagcia/barebones-nodejs/tree/taller>

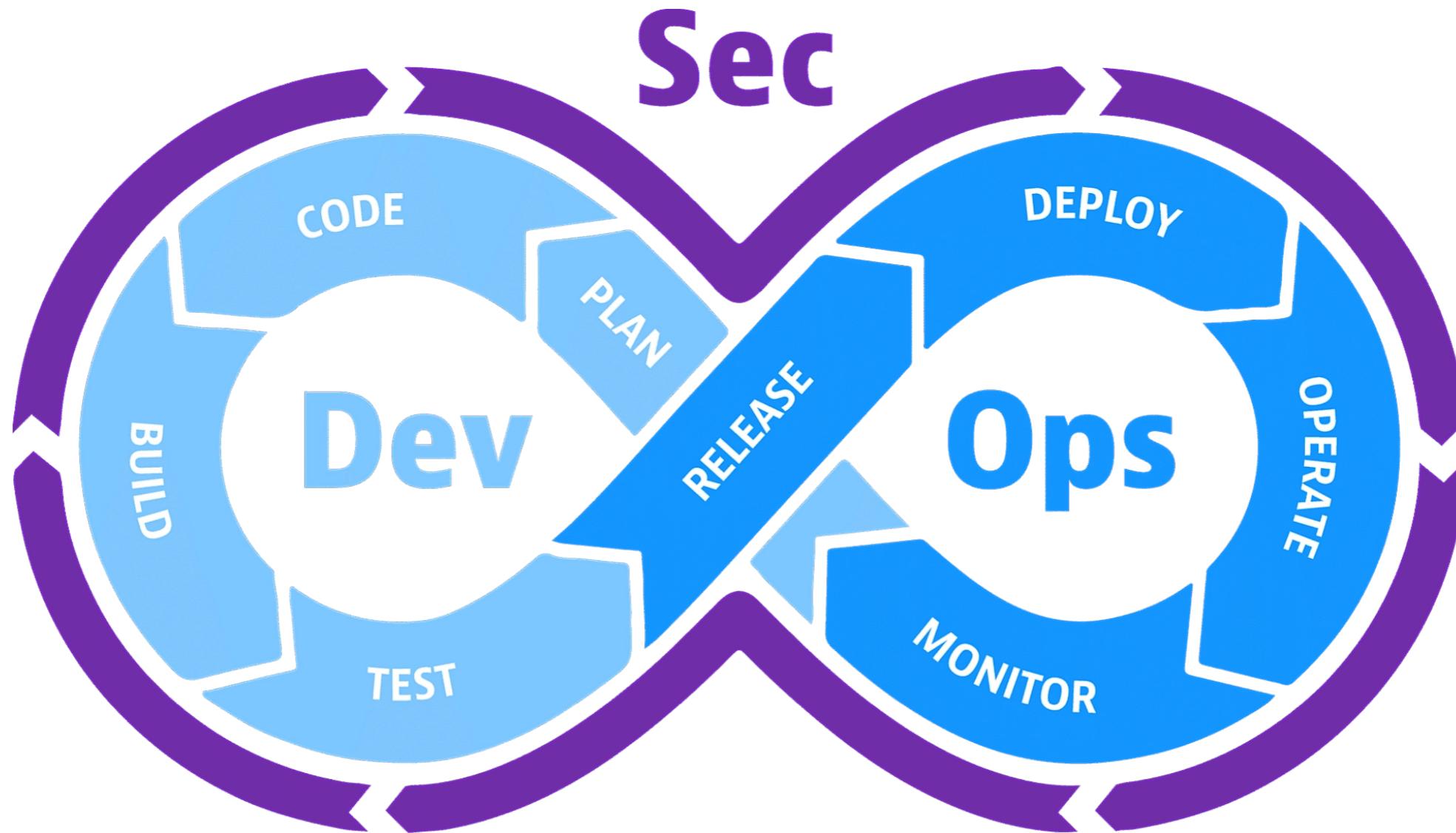


**Desarrollo de producto,
esa quimera**

DEVOPS



DEVOPS



DEV SECOPS



Infraestructuras



Buenas prácticas



Clean Code



Codificación segura



Testing y pruebas de código



Automatización

DevSecOps Stack

Gestión, colaboración y comunicación



Control de versiones



Automatización CI/CD



Análisis automático



Gestión de artefactos



Cloud/Infraestructura de despliegue



Orquestación de contenedores



Monitorización



Ciberseguridad



Operación



Desarrollo orientado a microservicios

- Programación monolítica:
 - Todo en un único paquete de software
 - Paradigma antiguo.
 - Difícil de mantener.
 - Difícil de parchear
 - Actualización con parada de servicio
- Programación orientada a microservicios
 - El paradigma actual
 - Especialización
 - Reutilización
 - Escalado selectivo
 - Actualización sin parada de servicio

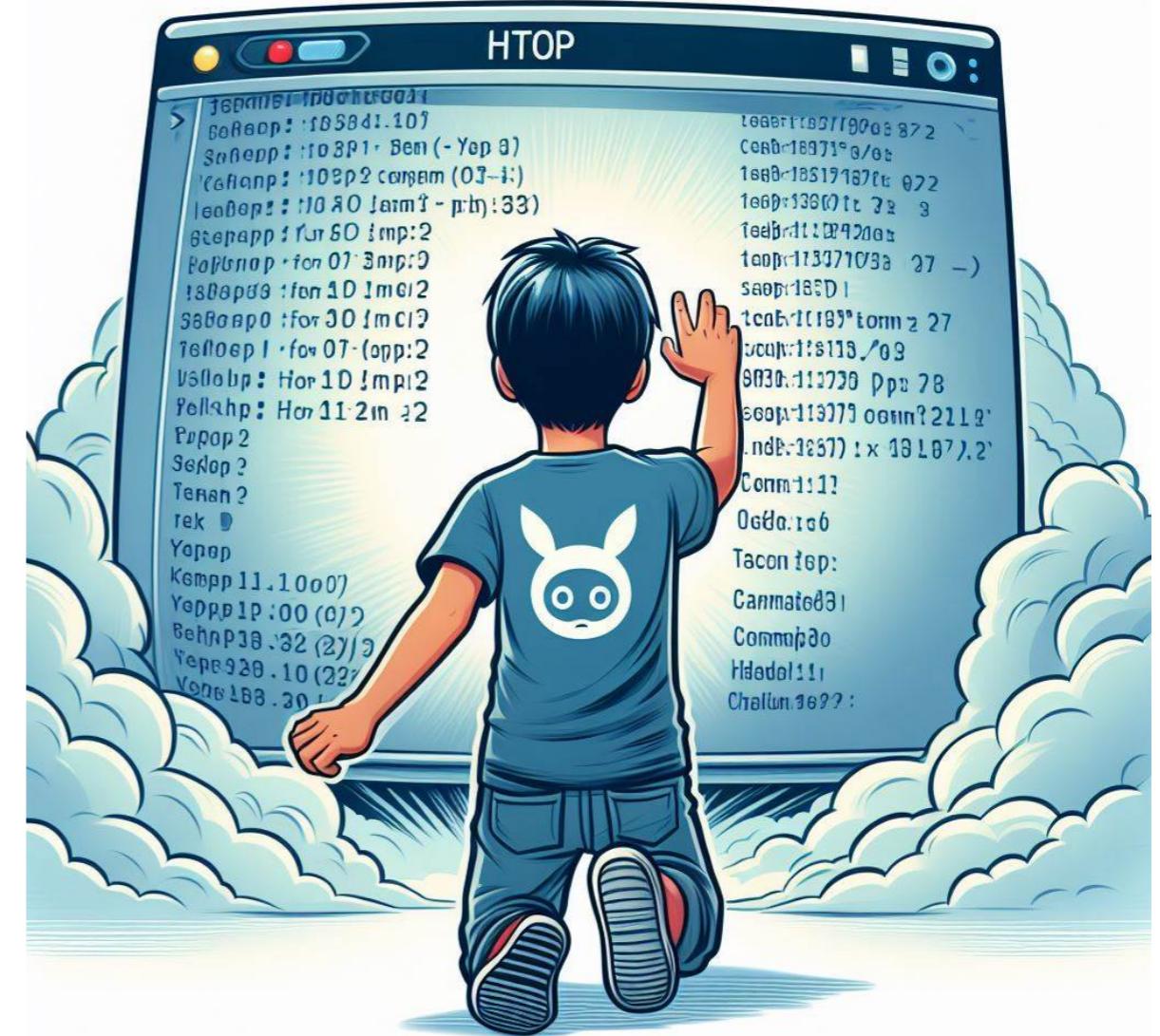


Contenedores

- Procesos aislados
- Su propio sistema de archivos
- Container IO
- Namespaces
- Cgroups
- Networking



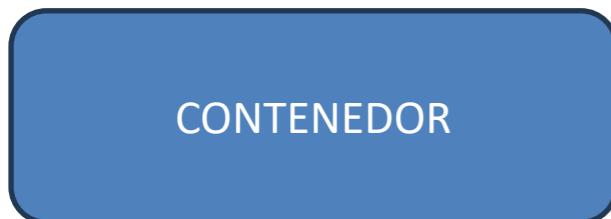
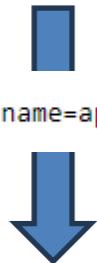
kubernetes



Imágenes de contenedores



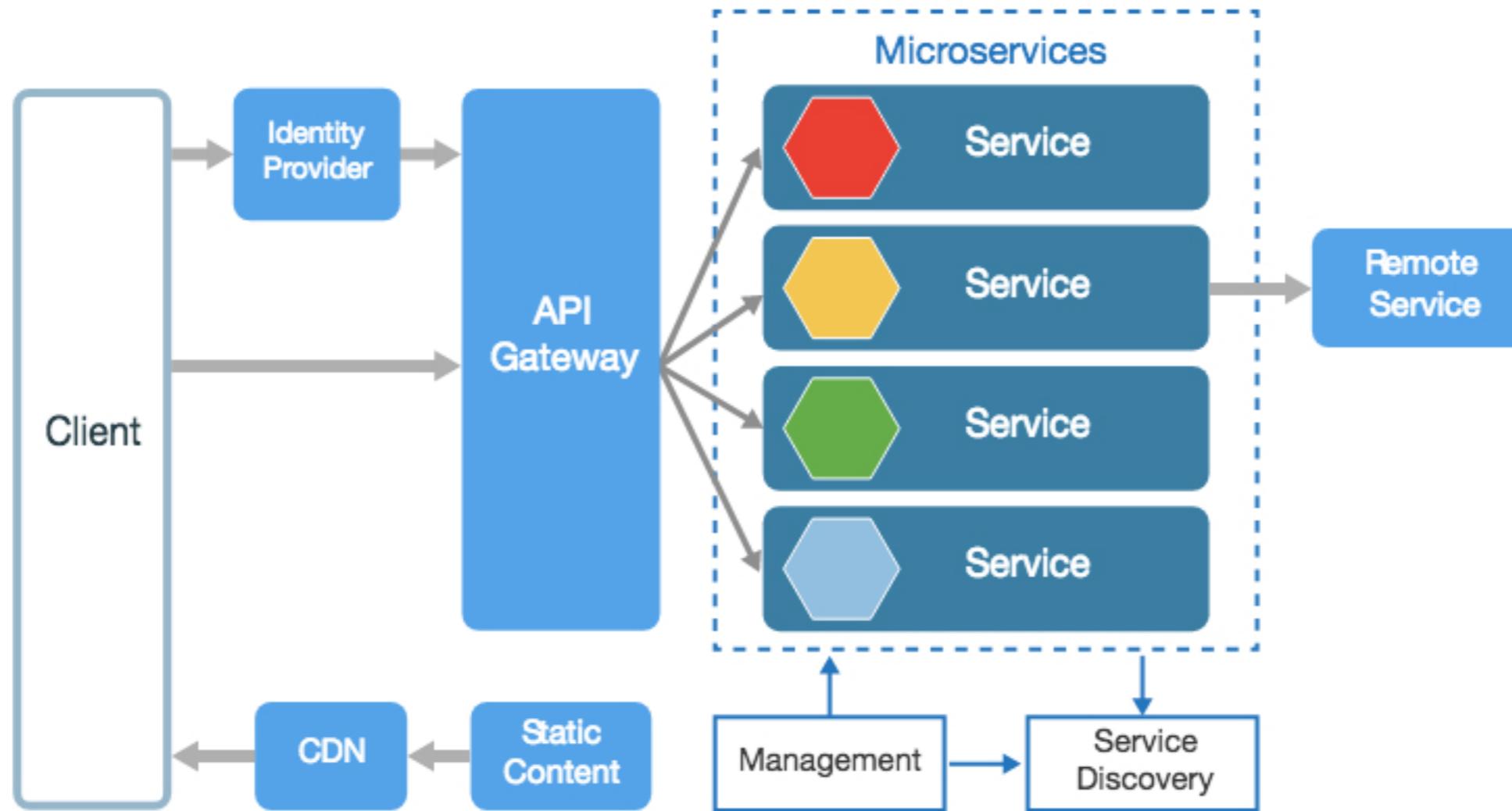
```
docker run --rm --name=app app-image:latest
```



```
1 # Use an official Node.js runtime as the base image
2 FROM node:14
3
4 # Set the working directory in the container to /app
5 WORKDIR /app
6
7 # Copy package.json and package-lock.json to the working directory
8 COPY package*.json .
9
10 # Install any needed packages specified in package.json
11 RUN npm install
12
13 # Copy the rest of the application to the working directory
14 COPY .
15
16 # Make port 3000 available to the world outside this container
17 EXPOSE 3000
18
19 # Run node-server.js when the container Launches
20 CMD ["node", "node-server.js"]
```

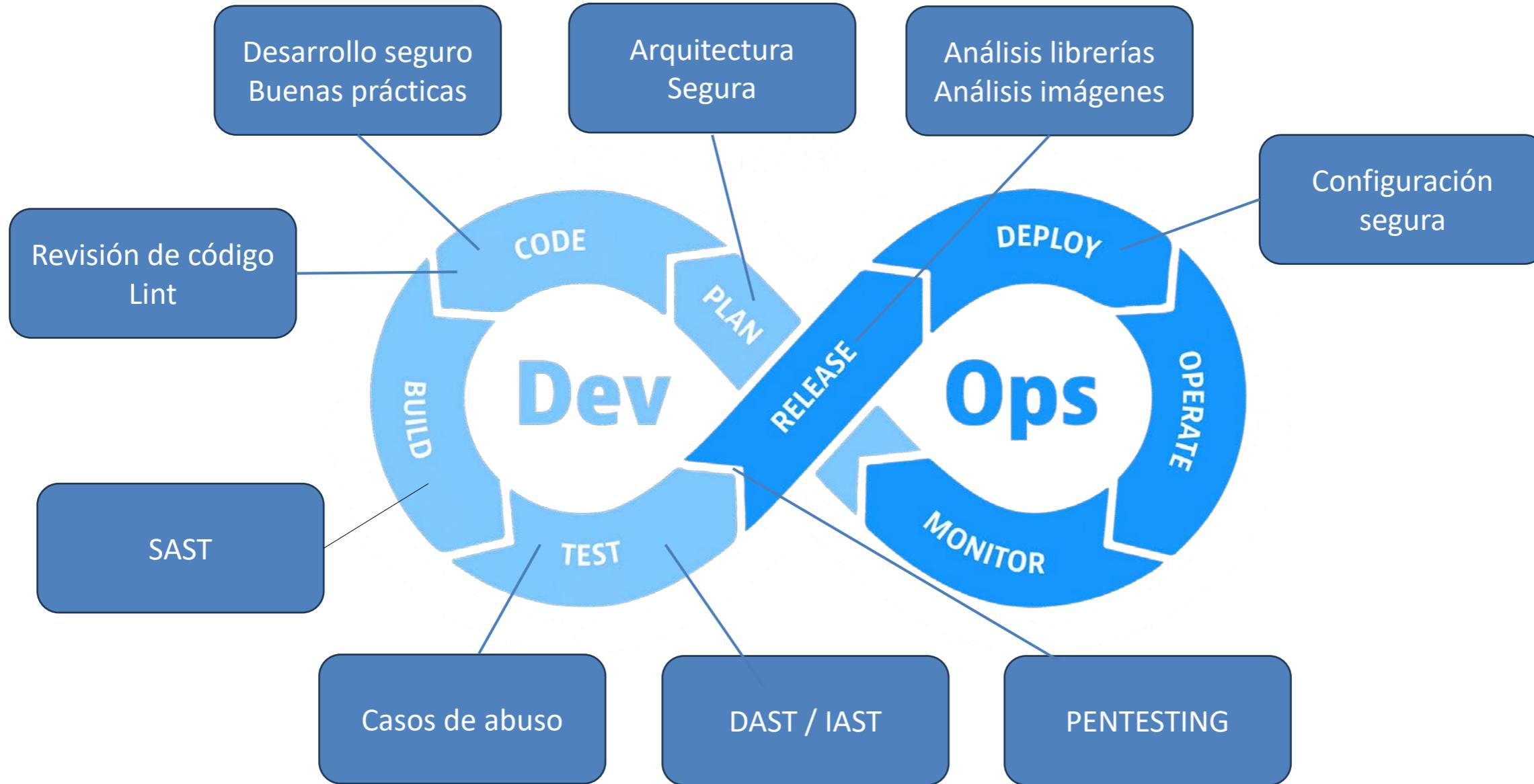
Caso práctico: Funcionamiento básico con Docker

Producto desarrollado en microservicios



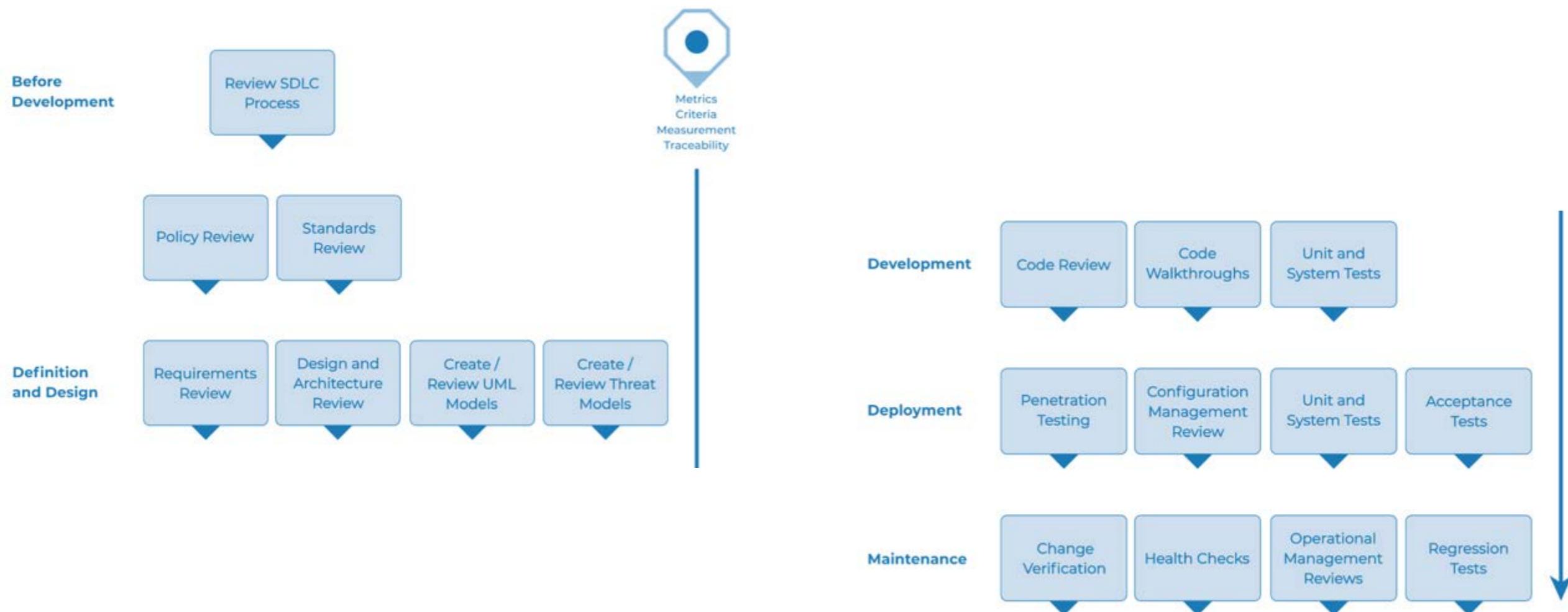
Seguridad en el proceso de desarrollo

Seguridad en el ciclo DEVOPS



Visión de OWSAP

Todos somos responsables de la seguridad



Clean Code

- KISS: lo más sencillo posible
- DRY: Evitar repetir sin motivo
- AGNI: Eliminar lo que no sea necesario
- Legible: Antes que conciso

Código WET:

```
//Variante A
let username = getUserName();
let password= getPassword();
let user = { username, password};
client.post(user).then(/*Variante A*/);

//Variante B
let username = getUserName();
let password= getPassword();
let user = { username, password};
client.get(user).then(/*Variante B*/);
```

Código DRY:

```
function getUser(){
  return {
    user:getUserName();
    password:getPassword();
  }
}

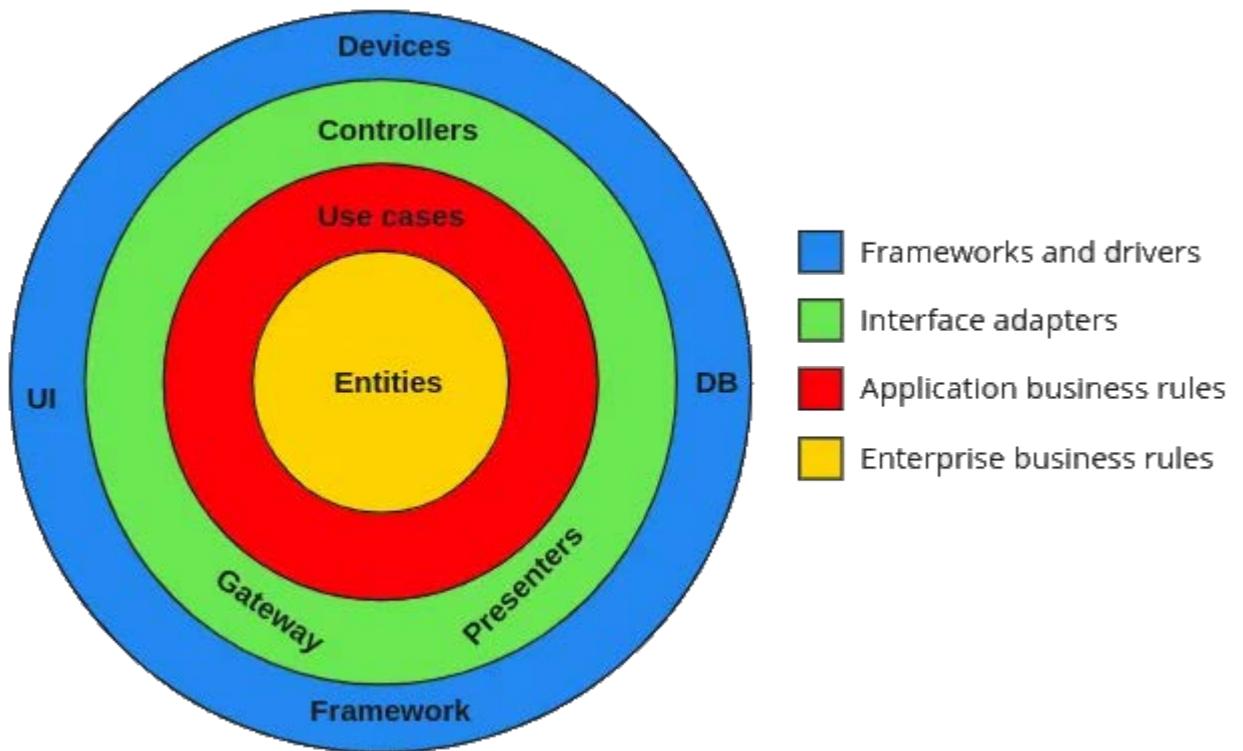
//Variante A
client.post(getUser()).then(/*Variante A*/ );

//Variante B
client.get(getUser()).then(/*Variante B*/);
```



Secure based Architecture

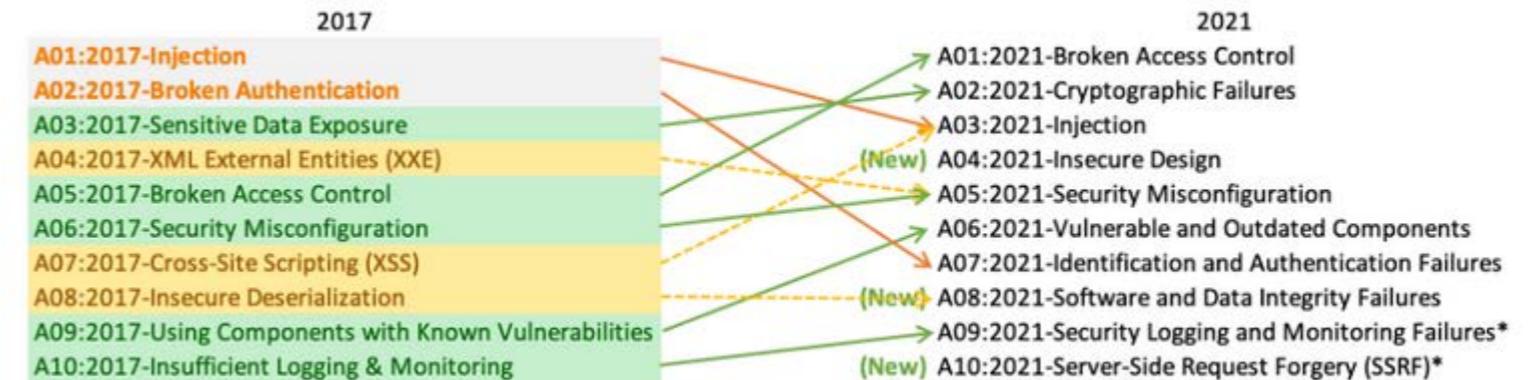
- Diseño desde la seguridad
- Establecimiento de una arquitectura que parta desde la seguridad
- Obligado cumplimiento
- Reglas de desarrollo



Desarrollo Seguro



TOP10



Desarrollo Seguro

<https://owasp.org/www-project-web-security-testing-guide/>

OWASP Web Security Testing Guide

Main [Release Versions](#) FAQ

Stable

View the always-current stable version at [stable](#).

[Unreleased 4.3]

[Version 4.2] - 2020-12-03

Version 4.2 introduces new testing scenarios, updates existing chapters, and offers an improved writing style and chapter layout.

[Download the v4.2 PDF here.](#)

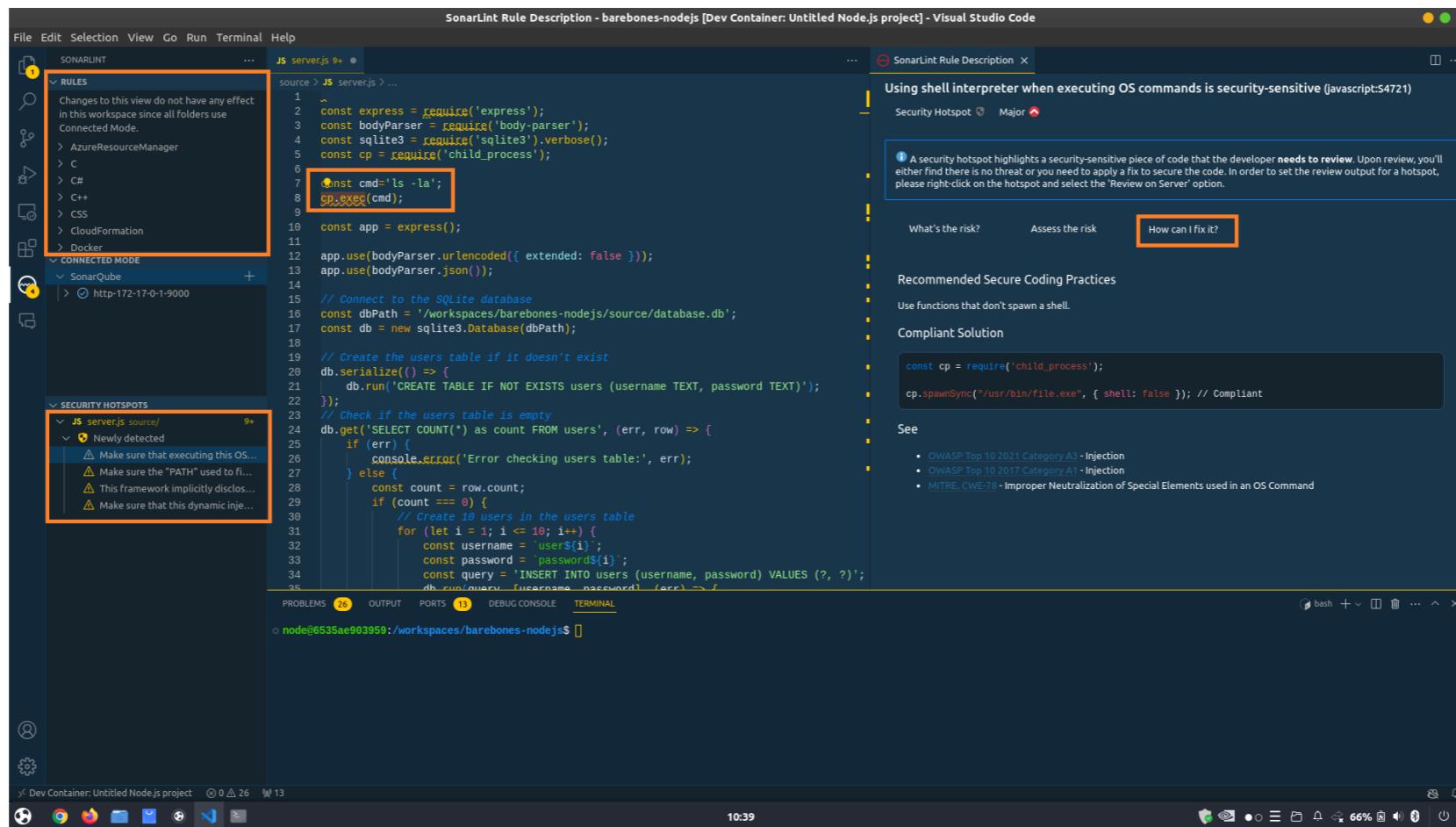
Desarrollo Seguro

- Respetar las buenas prácticas
- Directrices de desarrollo seguro
- Uso de librerías
- Uso de frameworks

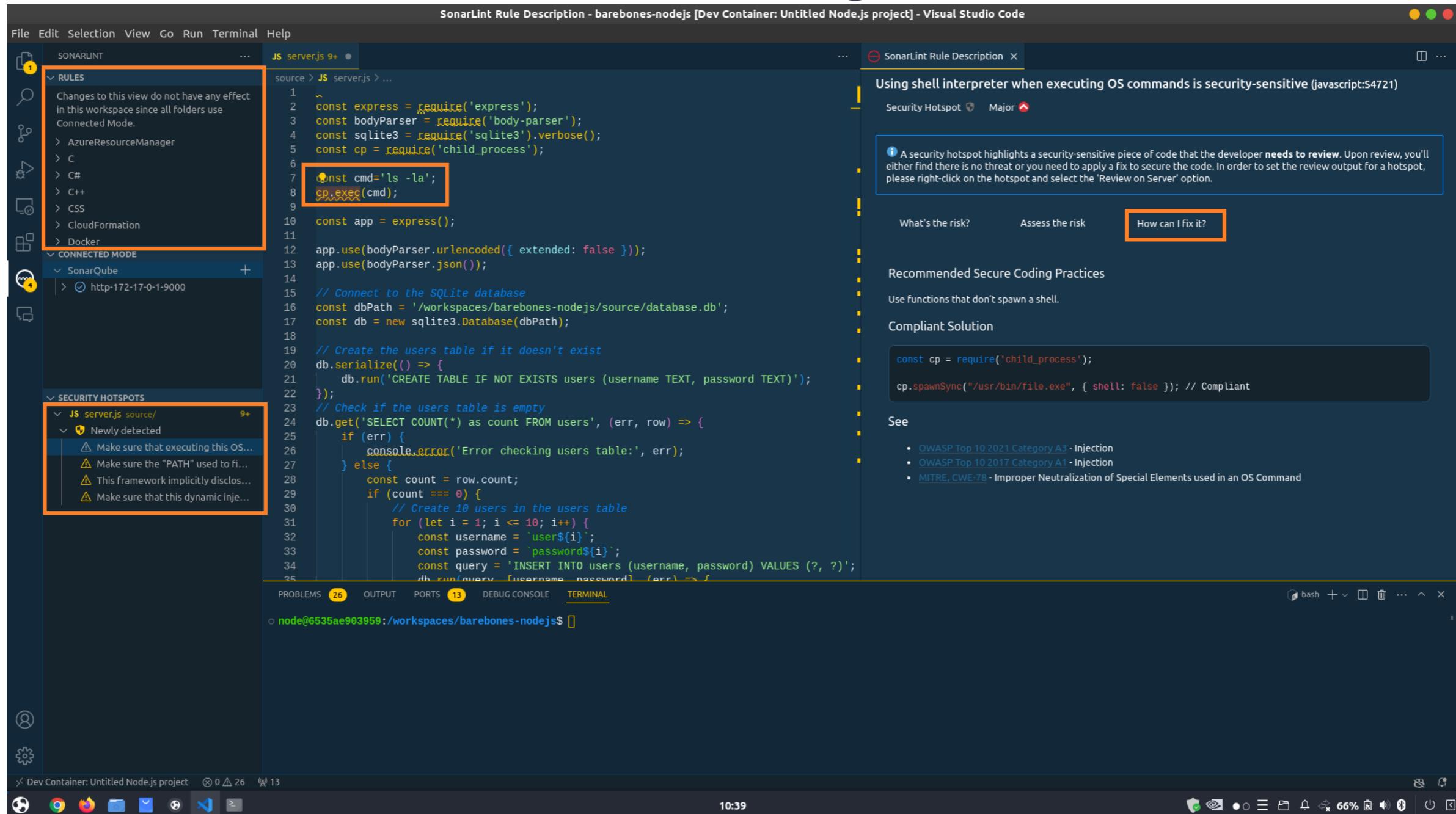


Revisión de Código: Lint

- Revisión de código: Cumplimiento de arquitectura
- Revisión de código: Cumplimiento de seguridad
- Lint: Automatización de revisión de código
- Falsos positivos

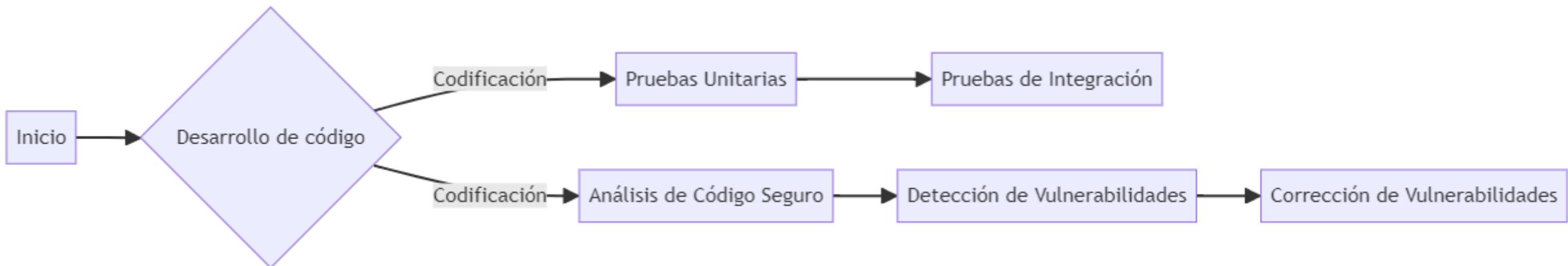


Revisión de Código: Lint



SAST

- Análisis estático de seguridad de aplicaciones
- Inspecciona el código fuente de una aplicación para detectar vulnerabilidades de seguridad, errores de codificación y otros riesgos de seguridad sin necesidad de ejecutar el programa
- Se basa en la definición y comprobación de reglas.



Caso práctico: Configuración y uso de SonarLint

CASOS DE ABUSO

- Son modelos de especificación para requerimientos de seguridad usado en la industria del desarrollo de software
- Especifican comportamientos deseados entre el software y otros productos involucrados, donde en general está estructurado



DAST/IAST

- DAST: (Análisis Dinámico de Seguridad de Aplicaciones) es una técnica de prueba que implica la evaluación de una aplicación en ejecución desde el exterior. DAST simula ataques contra una aplicación para identificar vulnerabilidades de seguridad que podrían ser explotadas por un atacante. Al interactuar con la aplicación a través de interfaces web o API, DAST evalúa la respuesta del sistema ante las entradas maliciosas, detectando problemas de seguridad en tiempo real mientras la aplicación está en funcionamiento.
- IAST: (Análisis Interactivo de Seguridad de Aplicaciones) combina aspectos de las pruebas SAST y DAST para proporcionar un análisis en tiempo real de las aplicaciones en funcionamiento. IAST se implementa como un agente dentro de la aplicación o el entorno de ejecución, permitiendo que observe el comportamiento de la aplicación y el flujo de datos durante las pruebas normales o la interacción del usuario, identificando vulnerabilidades de seguridad en contextos específicos de uso.

PENTESTING

- Son auditorías de seguridad de una aplicación.
- Utilización de herramientas
- Manual
- Nessus, OpenVAS, Nuclei, etc.



ANÁLISIS LIBRERÍAS / ANÁLISIS IMÁGENES

- Librerías con vulnerabilidades
- Imágenes de contenedores no seguras
- Análisis automático

```
node@6535ae903959:/workspaces/barebones-nodejs/source$ npm audit report
# npm audit report

semver 7.0.0 - 7.5.1
Severity: moderate
semver vulnerable to Regular Expression Denial of Service - https://github.com/advisories/GHSA-c2qf-rxjj-qqgw
fix available via `npm audit fix`
node_modules/simple-update-notifier/node_modules/semver
  simple-update-notifier 1.0.7 - 1.1.0
    Depends on vulnerable versions of semver
    node_modules/simple-update-notifier
      nodemon 2.0.19 - 2.0.22
        Depends on vulnerable versions of simple-update-notifier
        node_modules/nodemon

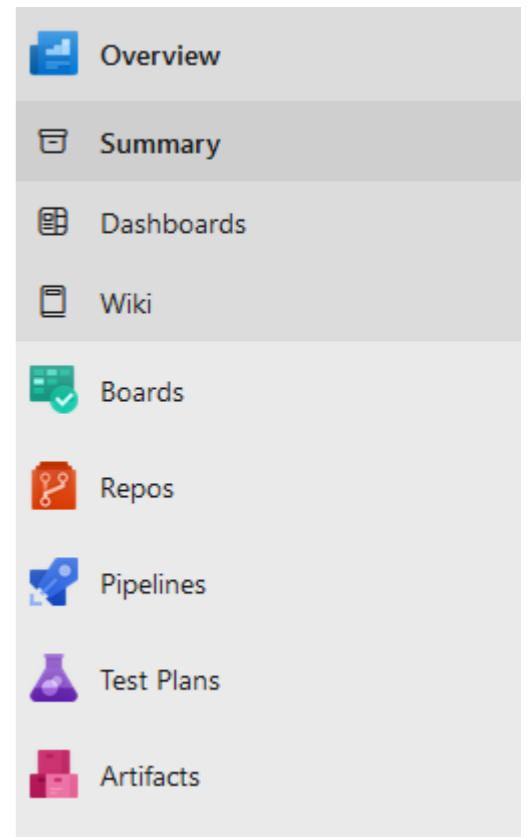
  3 moderate severity vulnerabilities

  To address all issues, run:
    npm audit fix
```

Caso práctico: Detección de vulnerabilidades

AZURE DEVOPS

Azure DevOps es un conjunto de herramientas en la nube de Microsoft diseñado para facilitar la colaboración entre equipos de desarrollo de software y agilizar el ciclo de vida de entrega de aplicaciones. Ofrece una amplia gama de servicios que cubren diferentes aspectos del desarrollo de software, desde la gestión del código fuente hasta la implementación y la supervisión de las aplicaciones en producción.



SUMMARY

A **automation-kubernetes**

Private  Invite 

About this project

Help others to get on board!
Describe your project and make it easier for other people to understand it.

+ Add Project Description



Project stats

Period: Last 7 days ▾

Repos	0
 Pull requests opened	0
 Commits by 0 authors	0

Members

3



Enter page title ×[Comandos comunes de kubectl](#)[Unfollow](#) 1[Edit](#)

Comandos comunes de kubectl

ss sdt.vs2 Visual Studio Subscription Just now

Obtener información básica del clúster

kubectl cluster-info: Muestra la información del clúster Kubernetes.

kubectl get nodes: Obtiene la lista de nodos en el clúster.

kubectl get namespaces: Obtiene la lista de espacios de nombres en el clúster.

Trabajar con recursos básicos

kubectl get <recurso>: Obtiene la lista de recursos, por ejemplo, kubectl get pods.

kubectl describe <recurso> <nombre>: Proporciona detalles detallados sobre un recurso específico, por ejemplo, kubectl describe pod mi-pod.

kubectl delete <recurso> <nombre>: Elimina un recurso específico, por ejemplo, kubectl delete pod mi-pod.

Trabajar con despliegues y servicios

kubectl create deployment <nombre> --image=<imagen>: Crea un despliegue con una imagen específica.

kubectl expose deployment <nombre> --port=<puerto>: Expone un despliegue como un servicio en un puerto específico.

kubectl scale deployment <nombre> --replicas=<replicas>: Escala un despliegue a un número específico de réplicas.

kubectl get services: Obtiene la lista de servicios en el clúster.

Ejecutar comandos en contenedores

kubectl exec -it <pod> -- <comando>: Ejecuta un comando interactivo en un contenedor, por ejemplo, kubectl exec -it mi-pod -- /bin/bash.

kubectl logs <pod>: Muestra los registros de un contenedor específico.

Gestión de configuración

kubectl apply -f <archivo>: Aplica la configuración definida en un archivo YAML o JSON.

kubectl create -f <archivo>: Crea recursos definidos en un archivo YAML o JSON.

kubectl delete -f <archivo>: Elimina recursos definidos en un archivo YAML o JSON.

Esta es solo una selección de los comandos más comunes de kubectl. Puedes encontrar una lista completa de comandos y opciones en la documentación oficial de Kubernetes y en la ayuda de kubectl.

0 visits in last 30 days

Comments

ss

Add a comment...

REPO

The screenshot shows a Git repository interface with the following details:

Repository Path: n-kubernetes / Repos / Files / automation-kubernetes

Branch: master

File List:

Name	Last change	Commits
automationdev	13 mar	514099da Organized folders jorge_garcia
automationv2	1 abr	d71c69bd Fix key osint and shodan automationv2
.gitignore	5 feb	2b10756d Add gitignore jorge_garcia
local-storage.yml	5 feb	271913b1 Reorganize estructure file apps jorge_ga
restart_app.sh	5 feb	7dd5c7d9 Add script rollout restart app jorge_garcia

Clone Repository Dialog:

- Command line:** HTTPS (https://n-kubernetes/_git/automation-kubernetes)
- Generate Git Credentials**
- IDE:** Clone in VS Code
- Note:** Having problems authenticating in Git? Be sure to get the latest version of [Git for Windows](#) or our plugins for [IntelliJ](#), [Eclipse](#), [Android Studio](#) or [Windows command line](#).

PIPELINES

- Compilation
- Release
- Publish

The screenshot shows the Azure DevOps Pipelines interface for the AdventureWorks project. The pipeline is titled "Enabling feature flags for Preview Attachment and Grid Views". It includes three parallel jobs: "Windows Job" (Running, 1m 53s), "Linux Job" (Running, 3m 29s), and "macOS Job" (Running, 3m 07s). The "Linux Job" is currently selected, showing its steps and logs. The steps are: Prepare job (< 1s), Initialize job (1s), Get sources (24s), Cmdline (28s), Nodetool (3s), and Install dependencies (2m 31s). The log output shows the command "yarn install v1.7.0" followed by several npm package resolutions and installations. The build was started on Sep 5 at 11:12am and has been running for 3m 29s.

Azure DevOps
Contoso / AdventureWorks Mobile / Pipelines / Builds / 10382

AdventureWorks

Overview

Pipelines

Builds

Releases

Library

Deployment groups

Project settings

Enabling feature flags for Preview Attachment and Grid Views

AdventureWorks/PackageFramework master #889

Release ...

Windows Job
Running 1m 53s

Linux Job
Running 3m 29s

macOS Job
Running 3m 07s

Linux Job
Agent: Hosted Linux

Started: Sep 5 at 11:12am
3m 29s

Prepare job < 1s

Initialize job 1s

Get sources 24s

Cmdline 28s

Nodetool 3s

Install dependencies 2m 31s

```
yarn install v1.7.0
$ node build/rpm/preinstall.js
[1/4] ⚡ Resolving packages...
[2/4] 🛡 Fetching packages...
[3/4] ⚡ Linking dependencies...
[4/4] 🚀 Building fresh packages...
$ rpm run compile
[#####
-----] 152/243
> code-oss-dev-build@1.0.0 compile ./adventureworks/build
> tsc -p tsconfig.build.json

⚡ Done in 4.89s.
$ node ./postinstall
[#] 2/2 removed './adventureworks/extensions/node_modules/typescript/lib/tsc.js'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.js'
removed './adventureworks/extensions/node_modules/typescript/lib/typescriptServices.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/typescriptFactory.js'
```

ARTIFACTS

Azure DevOps Contoso / AdventureWorksMobile / Artifacts / Packages

Packages | Connect to feed | Recycle bin

Package	Views	Source	Last pushed	Description
AdventureWorks.Framework Version 1.1.0		MyFeed	Yesterday	AdventureWorks core framework
adv-lib Version 1.3.3		MyFeed	2 days ago	libs for AdventureWorks front-end
com.adworks.app Version 2.0.2		MyFeed	2 days ago	AdvWorks Android app package
adventure-classifier-model Version 2.2		MyFeed	3 days ago	An object classifier for AdvWorks
adworks-build-tools Version 5.0.3		MyFeed	4 days ago	AdventureWorks development build tools
NUnit Version 3.11.0		NuGet Gallery	6 months ago	Unit testing framework for .NET
Newtonsoft.JSON Version 12.0.2-beta1	prerelease	NuGet Gallery	4 days ago	High-performance JSON framework for .NET
grunt Version 1.0.4		npmjs	13 days ago	The JavaScript task runner
express Version 4.16.4		npmjs	6 months ago	Fast, unopinionated, minimalist web framework
com.android.support.design Version 27.1.0		Maven Central	2 months ago	Add APIs for the Material Design specification
python-dateutil Version 2.8.0		PyPI	2 months ago	Extensions to the standard Python datetime module
numpy Version 1.16.2		PyPI	2 months ago	Fundamental package for array computing with Python

Project settings

AZURE DEVOPS End-To-End

1. Configurar un repositorio compatible con Azure Repos o externo.
2. Crear un proyecto en Azure DevOps.
3. Configurar un pipeline de compilación:
 - Seleccionar la fuente de origen.
 - Elegir una plantilla de Node.js.
 - Configurar los pasos de compilación.
4. Configurar un pipeline de implementación:
 - Configurar el artefacto de compilación como fuente.
 - Agregar un entorno de implementación.
 - Configurar los pasos de implementación.
5. Habilitar la integración continua y la entrega continua.
6. Ejecutar y supervisar los pipelines.
7. Publicar la aplicación después de una ejecución exitosa del pipeline de implementación.



Caso práctico: Azure DevOps publicación End-To-End

BASTIONADO



CLARA

Fichero Herramientas Opciones Ayuda

/clara>

CCN Versión 1.4

ens Esquema Nacional de Seguridad

Usuario: [REDACTED] Administrador

Equipo: [REDACTED]

Sistema operativo: Microsoft Windows Server 2012 Standard

Auditor: criptorafa

Organización: Steemlt

Unidad: spanish

Fichero de configuración: s\CLARA_ENS_1.4_x64\CLARA ENS x64\CLARA ENS\configurationENS.xml

Fichero IP's:

El archivo de configuración por defecto está cargado.
Puede analizar el sistema local ahora pulsando el botón "Analizar".
Por defecto, se realiza un nivel de análisis del nivel de cumplimiento de tipo ALTA.

Personalización > Analizar

Centro Criptológico Nacional

Nombre del sistema: [REDACTED]
Organización: Steemlt
Unidad: spanish
Categoría del sistema: ALTA

Auditado por criptorafa
Informes generados el día 18/09/2017 12:17:45
Versión de CLARA: 1.4

Datos del sistema

Mostrar todo

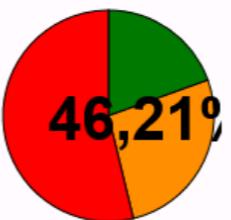
Mostrar

Análisis ENS

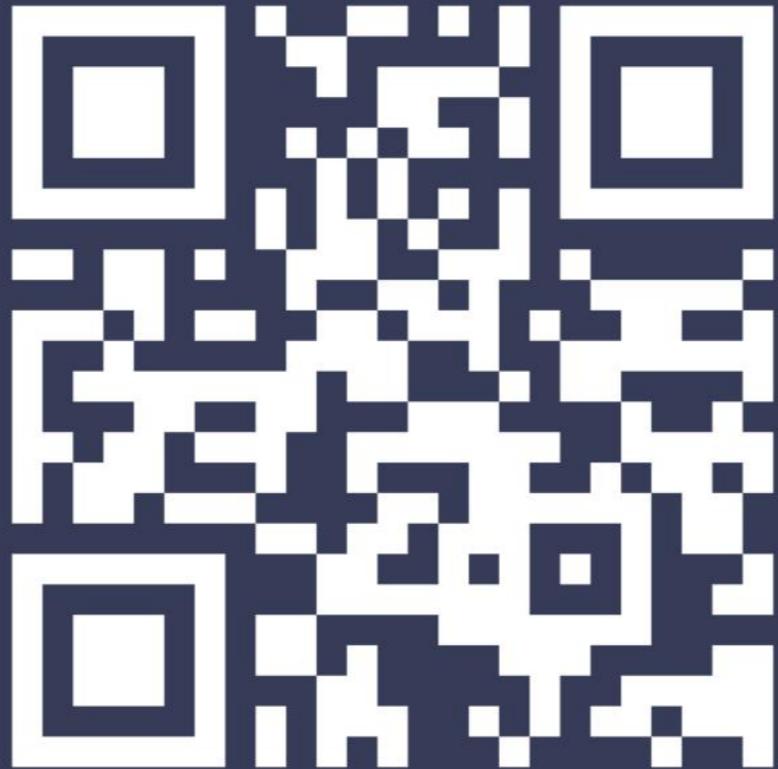
Ocultar

Resultados

Valor de criticidad Cumplimiento (46,21%)



Conoce más sobre nosotros:



izertis
Passion for Technology

izertis[.com](http://izertis.com)

