Trabajo Fin de Máster
Ingeniería de Telecomunicación

Seguridad en la integración continua de la metodología ágil y la filosofía DevOps

Autor: Eleazar Rubio Sorrentino

Tutor: Juan Manuel Vozmediano Torres

Dep. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Máster

Ingeniería de Telecomunicación

Seguridad en la integración continua de la metodología ágil y la filosofía DevOps

Autor:

Eleazar Rubio Sorrentino

Tutor:

Juan Manuel Vozmediano Torres

Profesor Titular

Dep. de Ingeniería Telemática

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Máster: Seguridad en la integración continua de la metodología ágil y la filosofía DevOps

Autor: Eleazar Rubio Sorrentino
Tutor: Juan Manuel Vozmediano Torres

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Apartado de agradecimientos.

Eleazar Rubio Sorrentino

Sevilla, 2017

Resumen

En los últimos tiempos de las empresas dedicadas al desarrollo de software como servicio (SaaS), los conceptos de metodología ágil y la filosofía Development and Operations (DevOps) están cobrando, cada vez más, un papel fundamental para el desarrollo de las mismas[8]. Según un estudio de alcance mundial realizado recientemente por CA Technologies, más del 75 por ciento de las organizaciones españolas coinciden en que las metodologías ágiles y DevOps son cruciales para el éxito de la transformación digital[11].

Este nuevo modo de entender el mundo del desarrollo de software (SW) posee una serie de elementos comunes, cada uno de ellos implementado con herramientas cada vez más conocidas y populares para las empresas que lo llevan a la práctica:

- Plataformas de desarrollo colaborativo y control de versiones de SW (por ejemplo GitHub), donde se almacena el código desarrollado por las mismas.

- Diferentes entornos o infraestructuras de trabajo para los desarrolladores, que van a permitir un desarrollo y despliegue continuo para las mejoras del producto: entornos de desarrollo, entornos de seguro de calidad o Quality Assurance (QA), preproducción, producción o entorno final, etc.

- Software de integración continua, en inglés Continuous Integration (CI), con las que automatizar los trabajos de despliegue de software.

- Plataformas basadas en el despliegue de contenedores (generalmente Docker) que encapsulan los distintos elementos que componen el producto final y optimizan los recursos utilizados en las máquinas que los contienen, en su mayoría subcontratadas a terceras compañías (Amazon Web Services, Microsoft Azure, etc.).

En el presenta Trabajo Fin de Máster (TFM) se pretende aportar e introducir al proceso comentado (y haciendo uso de las herramientas que este provee) una serie de análisis de seguridad que, ejecutados periódicamente, provean a la compañía de informes con los que poder identificar los siguientes problemas de seguridad durante el desarrollo de sus aplicaciones, siempre en continua integración con la línea de trabajo:

1. Si la aplicación creada tiene Vulnerabilidades (Common Vulnerabilities and Exposures (CVE)) en las librerías de dependencias de código utilizadas.

2. Si la imagen que se va a emplear para desplegar el contenedor de dicho SW contiene vulnerabilidades conocidas al nivel de Sistema Operativo (SO).

TODO - Cambiar enlaces por referencias bibliográficas.

Abstract

Lately inside Software as a Service (SaaS) companies, the agile methodology and DevOps philosophy concepts are taking a fundamental role for the development of them[8]. According to a recent global survey by CA Technologies, more than 75 percent of Spanish organizations agree that DevOps and agile methodologies are crucial to the success of digital transformation[11].

This new way of understanding the world of SW development has a various common elements, each of them implemented with well known and popular tools for the companies that put it into practice:

- Collaborative development platforms and SW version control systems (for example GitHub), where the code developed is keeped.
- Different environments and infrastructures for developers, which will allow to continuous development and deployment for product enhancements: development environments, Quality Assurance QA environments, pre-production, production or final environment, etc.
- Continuous integration CI software, with which to automate the software deployment.
- Platforms based on container deployment (usually Docker) that encapsulate elements that make up the final product and optimize the resources used in the machines that contain them, mostly subcontracted to third parties (Amazon Web Services, Microsoft Azure, etc.).

In the present master's thesis is intended to contribute and introduce to the process discussed (making use of the tools it provides), security analyzes that periodically executed provide the company with reports with which Identify the following security issues along of the process of development applications, always in continuous integration with the company pipeline:

1. If the application created has Vulnerabilities (CVE) in the code dependencies used.
2. If the image used to deploy the container of the mentioned SW contains known vulnerabilities at the Operative System (OS) level.

Índice

1 Introducción

You can ask 10 people for a definition of DevOps and likely get 10 different answers.

Dustin Whittle, 2014
Developer Advocate at Uber Developer Platform

En este primer apartado de la memoria se pretende realizar con el lector un recorrido a través del contexto que ha motivado el desarrollo del presente TFM para el Máster en Seguridad de la Información y las Comunicaciones de la Universidad de Sevilla, con el fin de aclarar el objetivo perseguido en su realización. Como conclusión al mismo, se definirá la estructura seguida durante la redacción, introduciendo cada uno de los apartados que se encontrarán a continuación.

1.1 Contexto y motivación

El año 2017, para empresas englobadas en todo tipo de sectores, está siendo el año de la transformación digital. Estos procesos de transformación exigen distintas formas de trabajo, más ágiles y colaborativas, con las que poder aplicar nuevas tecnologías que permitan conseguir los objetivos del negocio, en entornos que afrontan grandes desafíos culturales, organizativos y operativos e incluso pueden llegar a tener que lidiar con sistemas tecnológicos antiguos y casi obsoletos[2].

Es en este contexto donde el concepto DevOps empieza a sonar con más fuerza: el contexto de las metodologías ágiles.

De esta forma, DevOps es un concepto de trabajo, basada en el desarrollo de código, que usa nuevas herramientas y prácticas para reducir la tradicional distancia entre técnicos de programación y de sistemas, respondiendo a la necesidad experimentada por el sector tecnológico de dar una respuesta más rápida a la implementación y operación de aplicaciones. Este nuevo enfoque de colaboración que es DevOps permite a los equipos trabajar de forma más cercana, aportando mayor agilidad al negocio y notables incrementos de productividad.

Desde las pruebas de concepto hasta el lanzamiento, pasando por el *testing* y los entornos de prueba, todos los pasos involucrados requieren de la máxima agilidad posible (Figura 1.1), y eso pasa por integrar los procesos y los equipos de programación con los de sistemas[3].

Por otro lado, el concepto de contenedor de aplicación (aislamiento de espacio de nombres y gobernanza de recursos) a pesar de no ser un concepto novedoso, está cobrando cada vez más y más relevancia en el

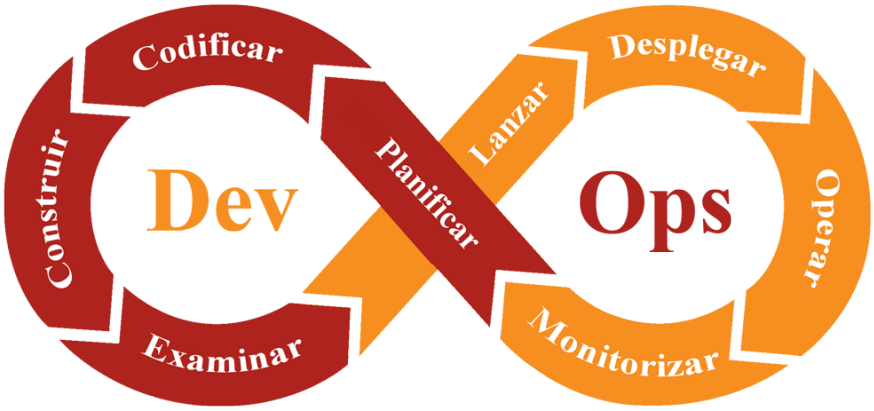

Figura 1.1 Introducción al proceso DevOps.

panorama de la empresa actual, de la mano de las continuas mejoras que experimentan las tecnologías que lo implementan, simplificando la administración y transformando la forma en que se desarrolla, distribuye y ejecuta el software, en forma de microservicio[1], además de proveer la habilidad de encapsular todo el entorno utilizado con el objetivo de ser desplegado en los sistemas de producción de la empresa, manteniendo las mismas características, aumentando la escalabilidad y disminuyendo notablemente los costes asociados a infraestructuras.

Los contenedores juegan un papel clave en un entorno DevOps porque soportan las implementaciones de la pila de desarrollo y operaciones completa y van en camino de formar parte de la definición básica de lo que se conocerá como DevOps en unos pocos años[4].

Además, la metodología DevOps representa una gran promesa a la hora de asegurar el desarrollo del software, ya que las organizaciones pueden potencialmente encontrar y remediar las vulnerabilidades con mayor frecuencia y al principio del ciclo de vida de la aplicación, ahorrando costes y tiempo. Conforme al informe de *"Seguridad de Aplicaciones y DevOps"* de octubre de 2016 promovido por Hewlett Packard Enterprise[5], que incluye tanto respuestas cualitativas como cuantitativas de profesionales de operaciones informáticas, líderes de seguridad y desarrolladores, se concluye que el 99 % de los encuestados confirma que la adopción de la cultura DevOps aporta la oportunidad de mejorar la seguridad de las aplicaciones. Sin embargo, solo el 20 % realizan análisis de seguridad de aplicaciones durante el desarrollo y el 17 % no utilizan ninguna tecnología que proteja sus aplicaciones, destacando una desconexión significativa entre la percepción y la realidad de la seguridad DevOps.

Es en el contexto planteado donde surge la idea del presente TFM: aportar mecanismos a la metodología DevOps que permitan analizar la seguridad de las aplicaciones desarrolladas y el contenedor que las albergará dentro de la infraestructura de la empresa, sin interferir de manera destructiva con el propio proceso de desarrollo y depliegue de la aplicación.

1.2 Objetivo

El objetivo del presente Trabajo Fin de Máster (TFM) es desarrollar un entorno, basado en contenedores, que pueda ser incluido en el proceso de desarrollo e integración continua de la empresa y con el que poder realizar tareas periódicas programadas para analizar estáticamente las posibles vulnerabilidades (CVETODO - ¿alguna más?) contenidas en las dependencias de aplicaciones desarrolladas mediante los

[1] Aproximación para el desarrollo software que consiste en construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros (normalmente una API de recursos HTTP)

lenguajes de programación Ruby y NodeJS, además de analizar a nivel del SO vulnerabilidades presentes en las imágenes que van a constituir el contenedor que dará soporte a dichas aplicaciones.

Como medio para alcanzar el objetivo planteado se va a hacer uso de una serie de aplicaciones y herramientas, entres las que cabe destacar(TODO - no me gusta esta última frase), que serán desarrolladas en el apartado 3:

- GitHub[7]: Plataforma de desarrollo colaborativo y control de versiones de SW donde almacenar, entre otras, el código desarrollado y que será analizadp.
- Jenkins[9]: Software de integración continua (CI) con el que automatizar los trabajos periódicos de análisis estático a realizar.
- Docker[6]: Proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, con el que desplegar los distintos elementos requeridos.

1.3 Estructura de la memoria

Para facilitar la lectura de la memoria actual, se cree conveniente presentar un resumen de cómo se estructuran los diferentes apartados que contiene.

En el apartado actual, Introducción, se presenta el TFM que se va a realizar, aclarando el objetivo perseguido, el contexto en que surge y los aspectos que motivaron su realización.

El apartado 2, Descripción de la técnica, se pretende dar a conocer, de manera objetiva, las características de la realidad representada, con rasgos tales como elementos que la componen, funcionalidad, utilidad, etc. En concreto, el apartado comienza describiendo el

El apartado 3, Entorno de trabajo, está dedicado a conocer las herramientas utilizadas para poder llevar a cabo el desarrollo y la implementación

El apartado 4, titulado Desarrollo de la aplicación, es el apartado principal de la memoria. En él, se detalla el proceso a seguir durante el desarrollo e implementación de la ”””””””””””””””””””””””” presentada en el proyecto, comenzando por los Primeros pasos a seguir, hasta llegar a presentar el Resultado final obtenido.

Por último, el apartado 5, está dedicado a las Conclusiones y evaluaciones surgidas del proyecto, los objetivos alcanzados y las líneas futuras de trabajo surgidas durante la realización de éste.

””””””””””””””””””””” PRESCINDIBLE ””””””””””””””””””””

Además, se advierte al lector que la mayor parte de herramientas y documentación utilizada para la realización del PFC se encuentran redactadas en inglés, incluyendo términos que en el ámbito del desarrollo de aplicaciones no poseen traducción literal al español. Es por esto, que en la redacción de la memoria actual no se utiliza traducción, ni letras cursivas, ni comillas para referir a dichos términos.

”””

TODO - Este apartado irá cobrando cada vez más y más sentido según avance la redacción de la memoria

2 Descripción de la Técnica

A good DevOps organization will free up developers to focus on doing what they do best: write software.

Rob Steward, 2015
Global Vicepresident at Verint-Systems.

Para comprender el desarrollo del trabajo aquí presentado, tal y como se ha llevado a cabo, se debe conocer la situación en que éste se desarrolla, la tecnología de la que se dispone y los elementos existentes y necesarios, de una manera objetiva.

Es por esto, que el apartado actual está orientado a conocer las características de la realidad representada y a introducir las bases tecnológicas del presente TFM, resaltando los conceptos más importantes.

2.1 Procesos de desarrollo en empresas TI

Pipelines y demás en empresas modernas... ¿Cómo se hacen las cosas? y comentar de qué forma no se va a interferir de manera destructiva en este proceso... y quizás esto último en otro apartado... ¿Incluyo aquí cómo puede ser ”más o menos” un día de trabajo DevOps?

2.2 (Software de) Integración Continua (Continuous Integration, CI)

2.3 OWASP

Breve introducción a lo que es, explicando que las herramientas utilizadas están recomendadas por ellos o, al menos, donde encontrar las recomendaciones de herramientas

2.4 CVE

2.5 Ruby y NodeJS. ¿Groovy?

Breve presentación a la importancia de estos lenguajes de programación, comentando que lo que se ha hecho aquí es extensible a otros lenguajes, con otras herramientas similares.
Concepto dependencias de código.

2.6 Análisis estático de dependencias de código

¿Qué es y cómo funciona?

2.7 Análisis estático de contenedores

¿Qué es y cómo funciona?

2.8 Comunicaciones seguras (SSH e intercambio de Tokens)

2.9 ¿Qué se ha hecho hasta ahora?

3 Entorno de trabajo

Technology is nothing. What's important is that you have a faith in people, that they're basically good and smart, and if you give them tools, they'll do wonderful things with them.

STEVE JOBS, 1994
BUSINESSMAN

Antes de comenzar el proceso de desarrollo de la aplicación es necesario preparar un entorno adecuado de trabajo, es decir, un conjunto de herramientas hardware y software que permitan llevar a cabo el proyecto con la mayor comodidad y precisión posible.

La correcta elección de un entorno de trabajo adecuado es fundamental a la hora de abordar cualquier tipo de proyecto, ya que el éxito o fracaso, o al menos la eficiencia del proceso de desarrollo del mismo, va a depender en gran medida de dicho entorno utilizado. El apartado actual presenta el entorno de trabajo utilizado para la realización de este TFM.

3.1 Git y GitHub

Los sistemas de control de versiones son programas cuyo principal objetivo es controlar los cambios producidos en el desarrollo de cualquier tipo de software, permitiendo conocer el estado actual de un proyecto, las personas que intervinieron en ellos, etc. Además, un buen control de versiones es tarea fundamental para la administración de un proyecto de desarrollo de software en general[1]. Git es uno de los sistemas de control de versiones más populares entre los desarrolladores, es gratuito, open source, rápido y eficiente, aunque gran parte su popularidad es debido a GitHub(Figura 3.2), un excelente servicio de alojamiento de repositorios de software que ofrece un amplio conjunto de características de gran utiilidad para el trabajo en equipo.

A continuación se muestran algunas de las características que han llevado a GitHub a ser tan valorado entre los desarrolladores[10]:

- Permite versionar el código, es decir, guardar en determinado momento los cambios realizados sobre un archivo o conjunto de archivos con la oportunidad de tener acceso al historial de cambios al completo, bien para regresar a alguna de las versiones anteriores o bien para poder realizar comparaciones entre ellas.

- Gracias a la gran cantidad de repositorios de SW públicos que alberga, es posible leer, estudiar

Figura 3.1 Logotipo de GitHub.

y aprender de el código creado por miles de desarrolladores en el mundo, permitiendo incluso la oportunidad de adaptarlos a las necesidades propias de cada desarrollador, sin alterar el original y realizando una copia o fork[1] de este.

- Tras haber realizado un fork de un proyecto y haber realizado algunos ajuste, introducido alguna mejora o arreglado algún problema que este pudiera contener, es posible integrar los cambios realizados al proyecto original (previa supervisión de su propietario, administrador o alguno de sus colaboradores), por lo que un repositorio puede llegar a ser construido mediante la contribución una gran comunidad de desarrolladores.

- GitHub posee un sistema propio de notificaciones con el que poder estar informado de lo que ocurre en torno a un repositorio concreto, ya sea privado a la compañía o público a la comunidad.

-

6. Visor de código

GitHub posee un estupendo visor de código mediante el cual, a través del navegador, podremos consultar en cualquier instante el contenido de archivo determinado, con la sintaxis correspondiente a el lenguaje en el que esté escrito. Este navegador es realmente rápido, y gracias a él podremos hacer pequeñas consultas o copiar porciones de código sin necesidad de bajarse todo el repositorio.

7. Mostrar tus conocimientos

Con Github puedes mostrar tus habilidades como desarrollador(a), puesto que es el código escrito en los archivos, donde reposa el resultado del proceso del desarrollo de software. Al compartir tu cuenta de Github con tu potencial empleador o cliente, este podrá ver la calidad del código que escribes a través de los proyectos públicos que estén en tu cuenta. Todos los proyectos que se escriben para ejecutar una idea, aprender un nuevo lenguaje o tecnología son válidos al momento de exhibir tus conocimientos, así que no dudes en publicarlo en tu cuenta. Como complemento a lo anterior, con Github Pages puedes crear incluso una página como esta que te sirva como portafolio, en la cual puedes escribir sobre ti, los conocimientos que posees o poner enlaces de los proyectos en los que has participado.

8. Registro de incidencias

Cada proyecto creado en Github incluye un sistema de seguimiento de problemas, del estilo sistema de tickets, este permite a los miembros de tu equipo (o a cualquier usuario de GitHub si tu repositorio es público) abrir un ticket escribiendo en este los detalles un problema que tenga con tu software o una sugerencia sobre una función que le gustaría que fuera implementada.

9. Compatibilidad

[1] Copia exacta en crudo del repositorio original que podrá ser utilizada como un repositorio git cualquiera

Github es una plataforma web, por tanto es independiente del sistema operativo que utilices, y además Git que es la herramient que si requiere instalación es compatible con todos los sistemas; Linux, OSX y Windows.

10. Precio

Github, es completamente gratis e ilimitado para proyectos públicos, es decir que todos podrán ver el código que estos contienen (aunque tu siempre tendrás el control sobre quien subirá cambios), sin embargo si deseas puedes tener proyectos privados adquiriendo uno de planes que ofrece, los cuales van desde 7 a 50 dólares mensuales, permitiendo crear 5 y 50 repositorios privados respectivamente.

3.2 Bundle Audit

3.3 NSP NodeJS

3.4 Clair (CoreOS)

3.5 Docker

3.6 Slack

3.7 Jenkins

Jenkins (Figura 3.2) es una herramienta autónoma de código abierto que puede utilizarse para automatizar todo tipo de tareas, como la construcción, prueba y despliegue de software. Jenkins puede ser instalado a través de paquetes de sistemas nativos, Docker, o incluso puede ser ejecutado de manera independiente en cualquier máquina con el entorno de ejecución de Java instalado[9].

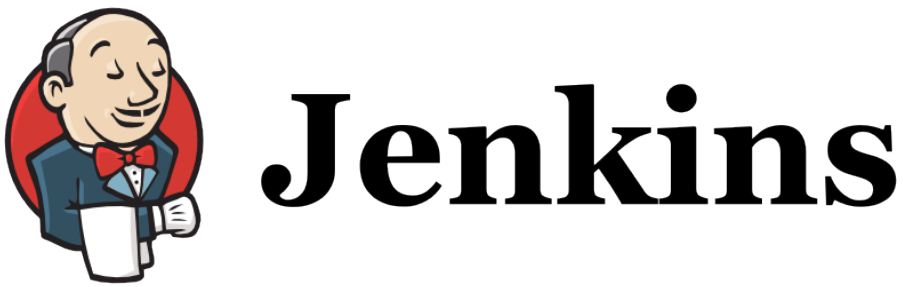

Figura 3.2 Integración continua (IC) con Jenkins.

Jenkins posee, entre otras, las siguientes ventajas:

- **Continuous Integration and Continuous Delivery**: As an extensible automation server, Jenkins can be used as a simple CI server or turned into the continuous delivery hub for any project.

- **Easy installation**: Jenkins is a self-contained Java-based program, ready to run out-of-the-box, with packages for Windows, Mac OS X and other Unix-like operating systems.
- **Easy configuration**: Jenkins can be easily set up and configured via its web interface, which includes on-the-fly error checks and built-in help.
- **Plugins**: With hundreds of plugins in the Update Center, Jenkins integrates with practically every tool in the continuous integration and continuous delivery toolchain. If a plugin does not exist, you can code it and share with the community.
- **Extensible**: Jenkins can be extended via its plugin architecture, providing nearly infinite possibilities for what Jenkins can do.
- **Distributed**: Jenkins can easily distribute work across multiple machines, helping drive builds, tests and deployments across multiple platforms faster.
- It is an open source tool with great community support.
- It provides continuous integration pipeline support for establishing software development life cycle work flow for your application.
- It also provides support for scheduled builds & automation test execution.
- You can configure Jenkins to pull code from a version control server like GitHub, BitBucket etc. whenever a commit is made.
- It can execute bash scripts, shell scripts, ANT and Maven Targets.
- It can be used to Publish results and send email notifications.

Pipeline - A user-defined model of a continuous delivery pipeline, for more read the Pipeline chapter in this handbook.
Debo poner también la imagen de una pipeline de trabajo con verdes y rojos.

4 Desarrollo de la solución

Epígrafe.

AUTOR, AÑO

Apartado Desarrollo de la solución.

5 Conclusiones

Security is not a line in the sand. Protecting your business, customers, citizens' data, should be always your number one priority

Dr. Werner Vogels, 2017
CTO at Amazon.com

Apartado Conclusiones.

Índice de Figuras

Índice de Códigos

Referencias

[1] Israel Alcázar, *Introducción a Git y Github*, Junio 2014, Enlace.

[2] Elena Arrieta, *DevOps: la tecnología y el negocio deben hablar el mismo idioma*, Mayo 2017, Enlace.

[3] Claranet, *DevOps: qué es y cómo lo aplicamos*, Accedido: Agosto de 2017, Enlace.

[4] Alan R. Earls, *Construir un entorno DevOps con microservicios y contenedores*, Diciembre 2015, Enlace.

[5] Hewlett Packard Enterprise, *Application Security and DevOps*, Octubre 2016, Enlace.

[6] Docker Inc., Accedido: Agosto de 2017, Enlace.

[7] GitHub Inc., Accedido: Agosto de 2017, Enlace.

[8] Consultor IT, *Estudio CA Technologies sobre la importancia de la Agilidad y DevOps en el desarrollo de software [pdf de 20 pgs.]*, Enero 2017, Enlace.

[9] Jenkins, Accedido: Agosto de 2017, Enlace.

[10] Erlinis Quintana, *10 razones para usar Github*, Junio 2015, Enlace.

[11] CA Technologies, *Accelerating Velocity and Customer Value with Agile and DevOps*, Enero 2017, Enlace.

Glosario

CI	Continuous Integration III, V
DevOps	Development and Operations III, V, 1, 2
OS	Operative System V
QA	Quality Assurance III, V
SaaS	software como servicio III, V
SO	Sistema Operativo III
SW	software III, V
TFM	Trabajo Fin de Máster III, 1–3