Estudio del estado del arte de las tecnologías de contenedores

Autor: Roca Blázquez, José Luis

Directores: Suárez Lorenzo, Fernando y Fernández Gavilanes, Milagros.

Contacto: jose_luis_roca@outlook.es

Resumen:

En este trabajo se ofrece un repaso del estado actual de la tecnología de contenedores. El texto pueda servir como introducción a este campo para personas que, teniendo unos conocimientos técnicos suficientes, no conocen ni han manejado nunca dicha tecnología. Se sigue una secuencia en la que se empieza por explicar los antecedentes históricos de los contenedores; se continúa explicando los fundamentos técnicos de bajo nivel sobre los que se asientan las distintas implementaciones existentes, y esto da pie para conocer las características principales de los contenedores y por qué han supuesto una revolución en el mundo del desarrollo y despliegue de aplicaciones. Se explican asimismo sus posibles usos, y dado que puede surgir la inquietud de comparar los contenedores con la virtualización, se realiza una comparación entre ambas tecnologías, indicando sus similitudes, diferencias y casos de uso.

Ha habido bastantes implementaciones de entornos de contenedores, y algunas de ellas, las más conocidas e importantes, se mencionan en este trabajo. Esto sirve además como argumento para la inclusión de varios intentos de normalización de algunos de sus aspectos en el capítulo dedicado a implementaciones.

En la parte final del trabajo se habla sobre infraestructuras de contenedores y la necesidad de organización mediante lo que se denomina “orquestación”. Se termina con algunos apuntes sobre el futuro de la tecnología de los contenedores, y se añaden las conclusiones y posibles líneas futuras para otros trabajos que puedan complementar a éste; los caminos posibles son muchos.

Palabras clave: Contenedores, microservicios, DevOps, orquestación, Docker, Kubernetes.

---

1. Introducción

Durante las últimas décadas ha habido una evolución en el desarrollo del hardware y del software que ha traído consigo cambios en la manera de implantar sistemas y servicios. Ya en la década de los años 60 del siglo XX se desarrollaban sistemas operativos que permitían homogeneizar el desarrollo de las aplicaciones en distintas plataformas hardware. Con el despliegue de redes de comunicaciones aparece el desarrollo de la provisión de servicios en red y éstos se sustentan en plataformas de servidores de distinta manera según evoluciona la tecnología: o bien había una correspondencia unívoca entre servicio ofrecido y servidor físico que lo soportaba, o se usaba un solo servidor para proporcionar varios servicios, o se usaban técnicas de virtualización, ya soportadas mediante nuevas capacidades incorporadas en las modernas CPU, para alojar varias máquinas virtuales en un servidor.

En distintas familias del sistema operativo Unix se llevan implementando desde hace varias décadas formas de ejecución de conjuntos de procesos aislados de otros procesos. Esta forma de trabajo sufrió un desarrollo grande con el añadido de determinadas características a los núcleos de los sistemas operativos, ya no solo de Unix sino también de Linux. Se acuña el término "contenedor" para denominar una unidad de ejecución independiente de otras unidades similares y controladas por un mismo sistema operativo. Esta estructura independiente, si se complementa con herramientas de gestión de usuario, soluciones de despliegue en distintos servidores y entornos de control de clusters, permite proporcionar servicios de una manera muy robusta y fiable.

2. Desarrollo

El núcleo de Linux dispone de una serie de características y capacidades que permiten aislar conjuntos de procesos del sistema operativo que los sustenta, y de otros conjuntos de procesos. Estas características, llamadas *cgroups*, *namespaces* y *capabilities*, son la base para conseguir desarrollar sistemas cada vez más complejos que aportan soluciones de contenedores que anteriormente no eran tan fáciles de obtener.

La posibilidad de disponer de contenedores como conjuntos de procesos con una gran independencia de otros conjuntos concurrentes en la misma máquina brinda enormes ventajas. Inmediatamente se aplica tal posibilidad para soluciones en las que anteriormente se utilizaba la virtualización. En el campo de la virtualización se hace uso de capacidades implementadas ya en las propias CPU y se usan máquinas virtuales que ofrecen a sus sistemas operativos independientes los recursos hardware que el hipervisor determina. El resultado son estructuras relativamente pesadas, pues cada máquina virtual debe contener todos los elementos de un servicio (hardware virtualizado, sistema operativo, aplicaciones, librerías y datos), aunque muy flexibles, ya que es posible implementar máquinas virtuales con distintos sistemas operativos instalados en un mismo servidor físico. La virtualización permite abstraer los sistemas operativos de las plataformas *hardware* subyacentes. Los contenedores, por otra parte, son estructuras más ligeras tanto en almacenamiento como en utilización de recursos físicos, si bien tienen algunas limitaciones en relación a las máquinas virtuales; por ejemplo, no se puede tener un sistema operativo completo distinto del sistema operativo base del servidor en el cual se trabaja. Haciendo un paralelismo con la virtualización pero a otro nivel, puede decirse que los contenedores permiten abstraer las aplicaciones de los sistemas operativos

subyacentes. Realmente la virtualización y los contenedores no son tecnologías antagónicas sino complementarias, y cada una de ellas tiene sus propios campos de aplicación. La virtualización permite tener sistemas operativos completos, sean cuales sean, sobre servidores que contienen un hipervisor que les brinda un hardware emulado. En algunos casos es la virtualización la única solución posible, por ejemplo cuando se desea mantener en funcionamiento una aplicación antigua desarrollada sobre un sistema operativo que ya no se mantiene y no se puede ejecutar en un *hardware* moderno. Por otra parte los contenedores conllevan unas enormes ventajas en el desarrollo y despliegue de aplicaciones. Tradicionalmente ha sido habitual codificar aplicaciones como estructuras monolíticas que daban un servicio externo final a base de implementar todas las relaciones entre los distintos elementos internos en un solo bloque. Sin embargo es relativamente cómodo y bastante eficiente desarrollar aplicaciones de manera que sus funciones internas se dividan en pequeños servicios independientes, y a su vez estos servicios se implementan mediante contenedores. Esta forma de desarrollo con microservicios proporciona flexibilidad y grandes posibilidades de desplegar servicios robustos con tolerancia a fallos y con resiliencia.

A lo anterior se une además la facilidad de despliegue de aplicaciones que se sustentan en contenedores a lo largo de una infraestructura compleja de servidores. Existen herramientas que integran las acciones de desarrollo-compilación y despliegue en clusters. La línea divisoria entre los equipos de desarrollo y producción se hace cada vez más fina, apareciendo el concepto de DevOps para referirse a la fusión entre desarrollo y operaciones, y esto facilita el trabajo de estas áreas y aporta una gran facilidad a la hora de solucionar errores, realizar cambios e implementarlos en una gran infraestructura que soporta distintos servicios. Los sistemas completos CI/CD (siglas por las que se conoce a los entornos de integración y entrega continuas, también desarrollo continuo) son también algunos de los más beneficiados por el desarrollo y perfeccionamiento de las distintas tecnologías de contenedores.

La teoría del aislamiento de conjuntos de procesos debe no obstante ser implementada de alguna manera efectiva y cómoda para que se pueda usar de manera práctica. En este sentido ha habido distintas soluciones en los últimos años. Las implementaciones implican algún añadido o modificación en algunos casos al sistema operativo base (como por ejemplo, módulos del kernel específicos para trabajar con contenedores), aunque no siempre es así. Deben incluir desde luego algún sistema de "imágenes", que son las estructuras de ficheros que se almacenan en repositorios y posteriormente se descargan en los servidores, y a partir de las cuales se construyen de manera dinámica los contenedores según se van creando y ejecutando. Además las implementaciones también deben tener en cuenta una serie de herramientas para poder operar de manera práctica y real con contenedores, creando imágenes, subiéndolas a un repositorio, descargándolas, ejecutando un contenedor, parándolo, volviéndolo a lanza, eliminándolo, proporcionando listados de imágenes descargadas y de contenedores en ejecución, etc.

Algunas implementaciones no han sobrevivido al paso del tiempo y han dejado de mantenerse, otras lo han hecho aunque no tienen mucho uso y no están demasiado extendidas. Hay una que ha tenido una gran aceptación y se utiliza ampliamente: *docker*. Esta solución proporciona todo lo necesario para trabajar con contenedores en entornos reducidos, por ejemplo en un solo servidor o en un cluster de pocos servidores, permitiendo también crear imágenes y subirlas a repositorios públicos gratuitos (y limitados, evidentemente). Lo interesante es que gracias a *docker* la tecnología de contenedores se ha dado a conocer ampliamente y se ha extendido muchísimo en relación a la situación

anterior a su existencia. Esto se une a las posibilidades ofrecidas por las características de los núcleos de los sistemas operativos de limitar los recursos del servidor (CPU, memoria, comunicaciones…) que se ofrecen a los contenedores albergados en la máquina física en cuestión, así como medir, y por tanto tarificar, el uso real de dichos recursos; en conjunto todo ello ha permitido que crezca enormemente el mercado en lo que se refiere a la posibilidad de contratar plataformas en las que alojar servicios basados en contenedores, al igual que ocurre con los sistemas equivalentes que utilizan la virtualización como servicio.

Los contenedores han facilitado desplegar aplicaciones de manera ágil y cómoda. Por ejemplo puede imaginarse la necesidad de desplegar en alguna organización un servicio de nube que permita a equipos de trabajo y a departamentos de cualquier organización realizar fácilmente trabajo colaborativo, y que ofrezca a los usuarios dados de alta en el sistema un espacio de almacenamiento con estrictos controles de acceso y compartición, sincronización de la información almacenada en dispositivos remotos, capacidades de comunicación mediante mensajes instantáneos y comunicación audiovisual en tiempo real, calendario y almacén de contactos accesibles mediante protocolos estándar, marcadores de navegadores, editores en línea de archivos ofimáticos estándar, acceso tanto desde interfaz web como desde aplicaciones para dispositivos móviles y más posibilidades más gracias a una arquitectura que permite la instalación de añadidos o aplicaciones. Con la tecnología de contenedores es extraordinariamente fácil realizar el despliegue de todos los servicios mencionados en cuestión de segundos. Todo lo anterior se consigue ejecutando una simple orden en una consola de un servidor donde, evidentemente, se haya realizado previamente la instalación de un entorno de gestión y ejecución de contenedores como *docker* (esta instalación, por otra parte, también es realmente sencilla). Una orden sencilla en *docker* descarga una imagen y ejecuta un contenedor que ofrece todos los servicios mencionados. Además, y como paso posterior en la demostración de las posibilidades de los contenedores, se puede realizar el mismo despliegue del mismo servicio de nube pero usando dos contenedores, dividiéndose cada uno de ellos las funciones. Esto es un ejemplo de microservicio. Este despliegue con más de un contenedor se define de manera muy sencilla en un fichero de texto con formato YAML (*Yet Another Mark Up Language*, formato de fichero ampliamente usado para describir estructuras de datos, configuraciones, formatos de mensajes, etc. y que en este caso se usa para la descripción de un despliegue de contenedores y servicios). El despliegue de estos microservicios según se describe en el mencionado fichero de configuración se realiza también con una simple orden en una consola.

Continuando con el punto de las implementaciones de contenedores y precisamente por el hecho de existir varias de ellas a lo largo del tiempo en el que se han desarrollado los contenedores, se ha hecho necesario proceder a normalizar o estandarizar algunos aspectos de los contenedores. La normalización siempre es un elemento positivo ya que permite desarrollar distintas soluciones que son interoperables y aporta flexibilidad para los administradores de los distintos sistemas a la hora de elegir una u otra solución.

Dado que el objetivo último del desarrollo de aplicaciones suele ser la provisión de servicios en red, es necesario tener en cuenta que lo habitual suele ser el despliegue de los distintos componentes que componen un servicio a lo largo de una infraestructura. Disponer de un servicio soportado por uno o más contenedores en un solo servidor físico no es la solución deseable, entre otras cosas por la posible limitación en el servicio ofrecido por las capacidades físicas del servidor y de los elementos asociados, y principalmente por la existencia de un único punto de fallo que puede suponer la caída total del servicio. Por tanto es necesario un paso más en el desarrollo e implementación de los contenedores,

contemplando el uso de clusters de servidores desplegados incluso en áreas geográficas muy extensas. En este punto es importante la gestión conjunta de toda esa infraestructura, de los elementos que la componen y de los servicios que soporta; es fundamental por tanto el concepto de "orquestación" como conjunto de recursos y herramientas que permiten una gestión integral y fiable. El ecosistema desarrollado por *docker* contempla el despliegue de contenedores en distintos nodos mediante una solución denominada *swarm* y además lo hace de una manera sencilla y eficiente. Sin embargo no se considera que sea una solución lo suficientemente robusta como para soportar servicios grandes en producción. Para ello se han desarrollado otros sistemas de orquestación, y el más conocido y extendido actualmente es *kubernetes*.

En la actualidad *kubernetes* es tanto un sistema utilizado *per se* para el despliegue de aplicaciones en infraestructuras complejas como también la base para otros productos, de código abierto y también comerciales, que sirven para gestionar todo lo relacionado con la provisión y comercialización de servicios de alojamiento de contenedores y despliegue de aplicaciones y servicios. Si se realiza una comparación en cuanto a las características y posibilidades entre la solución *docker swarm*, y *kubernetes*, se llega a la conclusión de que esta última es mucho más adecuada para infraestructuras especialmente grandes y para servicios que requieren una elevada disponibilidad y una robusta tolerancia a fallos. Naturalmente la potencia y las posibilidades tienen un precio en cuanto a la complejidad de diseño y gestión, que son superiores en el ámbito de *kubernetes* en comparación con el de *docker*.

El aspecto del almacenamiento es muy importante en despliegues complejos de servicios. Debe tenerse en cuenta que un contenedor es una estructura efímera que se crea en un servidor de manera dinámica a partir de una imagen, que no es más que un fichero con una determinada composición, en definitiva. Cuando un contenedor se destruye desaparece todo su contenido, y por tanto un contenedor de por sí no almacena nada de manera permanente, salvo que se haya implementado un sistema de asociación o enlace entre algunos directorios de un contenedor y una estructura de almacenamiento en un servidor. Tanto *docker* como *kubernetes* ofrecen formas de disponer de almacenamiento permanente que sobreviva al ciclo de vida de los contenedores que despliegen en forma de volúmenes, pero en despliegues reales de *kubernetes* suele recurrirse a soluciones de terceros que ofrecen una gestión muy mejorada del almacenamiento en clusters de servidores que albergan contenedores.

Las tecnologías de contenedores tienen un futuro bastante prometedor, pues son muchos los campos en los que pueden utilizarse. En muchas organizaciones es difícil dar el paso necesario para adoptar los contenedores como solución de despliegue, pues es necesaria una inversión, formación y tiempo de adaptación. A largo plazo no obstante es de esperar que se haga un uso muy extendido de los contenedores para distintos usos, como por ejemplo la investigación científica donde se requiere disponer de reproducibilidad, los sistemas de gestión de recursos y programación de trabajos o los entornos HPC (*High Performance Computing*) con aplicaciones de inteligencia artificial, *machine learning* y análisis de datos.

3. Conclusiones

Partiendo de unas capacidades aparentemente bastante simples en el núcleo de un sistema operativo y tras la implementación y el desarrollo de las herramientas adecuadas se consiguen unas funcionalidades y se posibilitan usos extremadamente potentes y flexibles de los contenedores. El

mundo del desarrollo software se ha visto beneficiado en gran medida por el concepto y la materialización de los sistemas de contenedores, y en el ámbito de la gestión del despliegue de aplicaciones y servicios se ha conseguido disponer de herramientas que facilitan enormemente el trabajo y que permiten integrar las áreas de desarrollo y operaciones en las empresas y organismos más complejos. El desarrollo en forma de microservicios se ha visto favorecido por la existencia de la tecnología de contenedores.

Actualmente hay dos implementaciones a distintos niveles que se han popularizado de manera especial: *docker* y *kubernetes*. La primera permite construir de manera relativamente sencilla imágenes y gestionarlas en almacenes o repositorios. La segunda puede hacer uso de forma fácil de las imágenes creadas y almacenadas con la primera, y se utiliza ampliamente en los despliegues de servicios en infraestructuras complejas con requisitos especialmente altos en cuanto a robustez y fiabilidad.