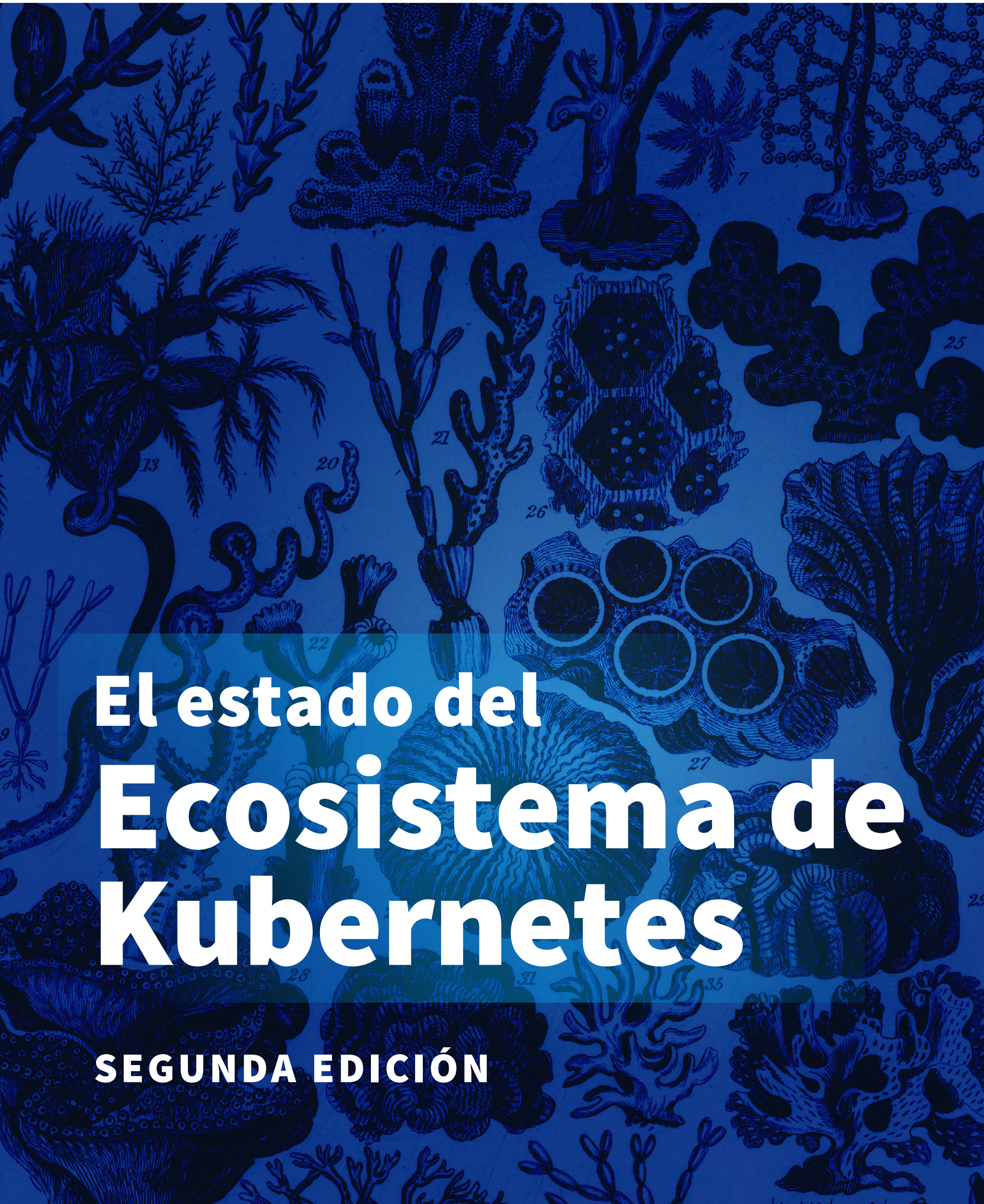
THENEWSTACK



El estado del Ecosistema de Kubernetes

SEGUNDA EDICIÓN

The New Stack

El estado del ecosistema de Kubernetes, segunda edición

Alex Williams, fundador y editor

Equipo de elaboración del libro electrónico:

Gabriel H. Dinh, productor ejecutivo

Janakiram MSV (Janakiram & Associates), autor

Jonathan Gold (Container Solutions), redactor técnico

Judy Williams, revisora de textos

Lawrence Hecht, analista de investigación asociado

Libby Clark, directora editorial y de marketing

Richard MacManus, editor de libros electrónicos

Sebastien Goasguen (TriggerMesh), redactor técnico

Equipo colaborador:

B. Cameron Gain, corresponsal en la UE

Benjamin Ball, director de Ventas y Gestión de Cuentas

Colleen Coll, responsable de Operaciones y Marketing de Medios Digitales

Eddie Rogers, productor de Medios Digitales

Jennifer Riggins, corresponsal en el Reino Unido

Joab Jackson, jefe de redacción

Michelle Maher, asistenta de redacción

20210205

THENEWSTACK

Contenido

Patrocinadores y socios

THENEWSTACK

Agradecemos el apoyo que nos han prestado los patrocinadores del libro electrónico:

DataStax es la empresa que está detrás de Apache Cassandra: una plataforma de gestión de bases de datos NoSQL sumamente escalable, de alta disponibilidad y nativa en la nube. DataStax brinda a usuarios y empresas la libertad de ejecutar sus datos en cualquier tipo de nube a nivel internacional, sin tiempo de inactividad ni ataduras contractuales.

Dynatrace es una empresa líder en inteligencia de software que desarrolla productos orientados a la nube empresarial. Es la única plataforma de inteligencia de pila completa basada en inteligencia artificial y completamente automatizada que ofrece información detallada sobre los ecosistemas de nube híbrida, caracterizados por ser muy dinámicos y por seguir los principios de la metodología de escala web. Por eso, las principales marcas del mundo confían en Dynatrace para ofrecer experiencias de usuario perfectas.

Harness, la plataforma de entrega de software, automatiza todo el proceso de integración y entrega continuas (CI/CD, por sus siglas en inglés), lo que permite a los ingenieros desarrollar, probar, implementar y verificar su software con los niveles de seguridad y velocidad que se le exigen a la empresa. Harness utiliza el aprendizaje automático para proteger su empresa cuando las implementaciones fallan. Entregar software nunca ha sido tan fácil.

Hewlett Packard Enterprise

HPE Ezmeral propone iniciativas de transformación digital traspasando tiempo y recursos de las operaciones de TI a las de innovación. Modernice y proteja sus aplicaciones, simplifique sus operaciones y utilice los datos para pasar de la información a la acción.

THENEWSTACK

KubeCon | CloudNativeCon

Las conferencias KubeCon + CloudNativeCon convocan a usuarios de la nube y tecnólogos para divulgar la informática nativa en la nube e impulsar su desarrollo. Organizadas por la Cloud Native Computing Foundation (CNCF), las conferencias KubeCon + CloudNativeCon son independientes de cualquier proveedor y cuentan con la participación tanto de expertos del sector como de los principales responsables de mantener conocidos proyectos como Kubernetes, Prometheus, Envoy, CoreDNS y containerd, entre otros.

PRISMA™ BY PALO ALTO NETWORKS

Un buen control de los accesos y una protección exhaustiva de los datos, las aplicaciones, los hosts, los contenedores y las tecnologías sin servidor: esta es la base necesaria para realizar la transición a la nube. Prisma, el paquete de seguridad en la nube más completo del sector, ayuda a los clientes a acelerar su proceso de migración a la nube gracias a sus tecnologías de visibilidad de riesgos y seguridad ininterrumpida.

Y nuestros socios:

Container Solutions

Container Solutions es una empresa de servicios profesionales especializada en la adopción de la nube nativa. Aunamos tecnología, cultura y estrategia para garantizar el éxito de la adopción. Container Solutions cuenta con oficinas en Ámsterdam, Londres, Berlín, Montreal, Praga, Varsovia y Zúrich.

TRIGGERMESH CLOUD NATIVE INTEGRATION PLATFORM

TriggerMesh proporciona una plataforma de integración nativa en la nube en tiempo real que le permite conectar servicios para automatizar los flujos de trabajo y mejorar la comunicación en el conjunto de la organización. TriggerMesh permite a los desarrolladores empresariales crear aplicaciones basadas en eventos y formadas por una combinación de servicios de varios proveedores de nube y sistemas locales.

Créditos

Janakiram MSV es el autor de este libro electrónico, analista jefe de Janakiram & Associates y profesor asociado del International Institute of Information Technology. También cuenta con las certificaciones Qualified Cloud Developer de Google, Amazon Certified Solution Architect, Amazon Certified Developer, Amazon Certified SysOps Administrator y Microsoft Certified Azure Professional. Janakiram es embajador de la Cloud Native Computing Foundation, y una de las primeras personas en haber obtenido los certificados Certified Kubernetes Administrators y Certified Kubernetes Application Developers. Ha trabajado para Microsoft, AWS, Gigaom Research y Alcatel-Lucent.

Lawrence Hecht ha contribuido a este libro electrónico en calidad de analista de investigación asociado y lleva 17 años elaborando informes sobre los mercados de las tecnologías de la información. Anteriormente, dirigió encuestas de satisfacción del cliente para 451 Research y TheInfoPro sobre mercados tecnológicos B2B, como el de la computación en la nube, el análisis de datos y la seguridad de la información. En 1999, estableció la Internet Public Policy Network (IPPN), una red de expertos que ofrecía servicios personalizados de investigación, elaboración de informes técnicos y consultoría en materia de políticas públicas de tecnología. Lawrence colabora con escritores para identificar datos que permitan desarrollar información estratégica basada en investigaciones primarias y secundarias. En la actualidad, crea contenido para programas de liderazgo de opinión y «evangelización» entre colegas y compañeros de profesión.. Lawrence se graduó por la Universidad de Rutgers con un *Bachelor of Arts* y cuenta con un Máster en Políticas Públicas de la Universidad de Georgetown.

Introducción

Hace tan solo cinco años, en junio de 2015, la gente no terminaba de entender ni el nombre ni el concepto de Kubernetes, un proyecto que había nacido el año anterior en el seno de Google.

La palabra «kubernetes» proviene del griego antiguo y significa «timonel de buque». Era una arquitectura tecnológica que apelaba a un cambio que, ya en 2015, llevaba varios años gestándose, con el aumento de empresas que operan a escala web. Docker seguía gozando de popularidad, pero la comunidad empezaba a explorar nuevas ideas en torno a la orquestación de contenedores. En junio de 2015, se anunciaba la fundación de una organización llamada Cloud Native Computing Foundation (CNCF), donde Kubernetes encontró un nuevo hogar.

Surgían, así, las arquitecturas con escalabilidad horizontal. Docker fue el primer megaproyecto de código abierto. Kubernetes siguió sus pasos, y no tardó en convertirse en el movimiento que definió la intersección entre la computación distribuida, las comunidades de código abierto y el avance de las arquitecturas de aplicaciones. Rápidamente se erigió como uno de los proyectos de código abierto más importantes de los últimos 20 años y algunos llegaron, incluso, a considerarlo la versión en la nube de Linux.

En 2017, con el fin de documentar el movimiento, The New Stack publicó nuestro primer libro electrónico acerca de la comunidad de Kubernetes y el ecosistema nativo en la nube que la acompañaba.

Desde 2018, la comunidad ha cambiado tanto y se ha desarrollado tan rápido que la primera edición empezaba a quedarse obsoleta. Necesitaba una revisión y una nueva perspectiva que reflejara la realidad de la comunidad de Kubernetes actual y su futuro inmediato.

Según un estudio llevado a cabo en 2019, casi el 80 por ciento de las empresas encuestadas por la CNCF ya utilizan Kubernetes en sus entornos de producción.

El éxito de convocatoria de las conferencias KubeCon + CloudNativeCon, que en Norteamérica pasaron de aproximadamente 1000 asistentes en 2016 a 4000 en 2017, da cuenta de la popularidad de Kubernetes y las tecnologías nativas en la nube. El pasado otoño, la conferencia KubeCon que

se celebró en San Diego reunió a más de 10 000 asistentes. En 2020, la KubeCon será virtual, por lo que se espera que la convocatoria sea mucho más multitudinaria.

En el momento de publicar este libro, todo el mundo está en casa y la COVID-19 obliga a que la participación en proyectos de código abierto sea un ejercicio virtual. Pero Kubernetes sigue creciendo... y su consolidación también.

Actualmente, la capitalización bursátil de la CNCF, integrada por 570 empresas, asciende a los 17,8 billones de dólares. Se trata de un agregado total de empresas que da una idea del volumen de inversión de la comunidad en general. A lo largo de todo este tiempo también ha habido adquisiciones significativas. En enero de 2018, Red Hat compró por 250 millones de dólares CoreOS, la empresa de software a la que debemos la solución de tiempo de ejecución del contenedor Tectonic. A finales de 2019, VMware compró Heptio, la empresa fundada por Joe Beda y Craig McLuckie, directores del equipo de Google que desarrolló Kubernetes junto con Brendan Burns, quien para entonces también había abandonado Google para trabajar en Microsoft. La compra de Heptio se cifró en 450 millones de dólares. En mayo de 2019, Palo Alto Networks adquirió Twistlock, una empresa de seguridad para contenedores, en una operación valorada en 410 millones de dólares. Ese mismo mes, VMware adquiría Bitnami. A principios del año pasado, VMware anunciaba la compra de Pivotal, la compañía responsable de las negociaciones de la comercialización de la plataforma como servicio de código abierto Cloud Foundry. En julio de 2020, justo antes de la publicación de este libro electrónico, SUSE adquirió Rancher Labs por un valor declarado de 600 millones de dólares.

El crecimiento de la comunidad, junto con la evolución de los microservicios como modelo de arquitectura de desarrollo, ha conducido a un conocimiento más profundo de las tecnologías de orquestación de contenedores y de la computación distribuida en general.

Ahora se entienden mejor los motivos por los que Kubernetes requiere una reserva subyacente de recursos de computación, almacenamiento y redes que puedan ejecutarse a través de una infraestructura local o de un servicio en la nube. Aun así, la arquitectura central subyacente resulta útil para explicárselo a los usuarios principiantes o más experimentados.

En este libro electrónico, The New Stack presenta los aspectos fundamentales de Kubernetes.

El objetivo es dar a conocer los conceptos básicos de planos de control, nodos de trabajo, detección de servicios, almacenamiento y redes. Analizaremos el modo en que Kubernetes proporciona las tecnologías de computación a escala web, los patrones de adopción y su función como aspecto universal de la infraestructura principal de una empresa.

El segundo capítulo del libro electrónico está dedicado a las plataformas de gestión de contenedores. Los proveedores y servicios en la nube ofrecen multitud de plataformas, pero todas ellas tienen algo en común: Kubernetes. Este libro electrónico explica por qué Kubernetes no solo es la arquitectura subyacente de las plataformas diseñadas para los centros de datos, los servicios en la nube y los modelos híbridos de las empresas, sino también para el perímetro.

El segundo capítulo describe, asimismo, qué función desempeñan las tecnologías nativas en la nube en las plataformas de contenedores. Repasamos por qué las tecnologías nativas en la nube son, en esencia, tecnologías de contenedor desarrolladas para funcionar en infraestructuras basadas en contenedores y, en particular, Kubernetes.

Los microservicios utilizan componentes que se ejecutan en varios contenedores, pero ¿cómo se gestionan las aplicaciones? Es una de las grandes cuestiones a las que se siguen enfrentando los desarrolladores. Las aplicaciones constan de distintos componentes, y el reto consiste en unir todos esos componentes de tal manera que no solo sea posible crear la aplicación, sino también gestionar el servicio a lo largo de todo su ciclo de vida.

El autor del libro electrónico, Janakiram MSV, sostiene que el modelo de contenedor como servicio (CaaS, por sus siglas en inglés) es lo que define ahora mismo la tecnología nativa en la nube. Pero es un modelo complejo y difícil de gestionar. Mírelo así: una arquitectura de Kubernetes en un entorno de producción es como una colmena. Es caótica, pero poner orden es posible si se siguen unas prácticas de ingeniería que permitan ejecutar los microservicios en contenedores orquestados por Kubernetes. Quienes lo han hecho han logrado desarrollar y gestionar sus aplicaciones a gran escala. Prosperan. Pero la realidad es que casi todo el mundo está aún al principio del proceso.

Ese es un buen resumen del desafío al que se enfrenta la comunidad de Kubernetes tras su prometedor comienzo, para seguir desarrollando una plataforma que sirva tanto a las operaciones

de las grandes empresas como a las de las pymes.

El mayor obstáculo es la cultura. Los desarrolladores necesitan encontrar la forma de empaquetar fácilmente sus aplicaciones y urge comprender mejor a quién le corresponde configurar los servicios. La política y la seguridad son otros dos asuntos importantes que dependen, en gran medida, de prácticas tradicionales.

Kubernetes, como algunas partes del ecosistema que aún están dando sus primeros pasos, tiene un futuro prometedor. Los datos, en Kubernetes, siguen siendo un espacio por definir; si Kubernetes es, ahora mismo, el plano de control por excelencia, el plano de los datos está todavía por resolver. Y, si bien las funciones de malla de servicios permiten gestionar, proteger y observar mejor el tráfico, la curva de aprendizaje sigue siendo demasiado pronunciada.

Pero son los conceptos asociados a la observabilidad lo que nos suscita un mayor interés. La supervisión es una práctica que fue particularmente adecuada para la arquitectura empresarial local. Está hecha para reaccionar, pero no podemos seguir reaccionando. Lo que el mundo espera de nosotros es que observemos lo que está pasando, nos anticipemos y demos con las respuestas.

Gracias por descargar este libro electrónico. Quedamos a su disposición para lo que pueda necesitar.

Alex Williams

Fundador y editor

The New Stack

@alexwilliams

alex@thenewstack.io

CAPÍTULO 1

Kubernetes, el sistema operativo para la nube

Durante los últimos diez años, el mercado de las infraestructuras de TI ha vivido un auténtico terremoto. En primer lugar, la infraestructura como servicio (IaaS, por sus siglas en inglés) revolucionó la forma en que las empresas aprovisionaban y consumían los recursos de computación, almacenamiento y redes. En la segunda mitad de la pasada década, asistimos al auge de los contenedores, las plataformas de gestión de contenedores y las soluciones de contenedores como servicio (CaaS) en la nube.

Los servicios de infraestructuras en la nube —como Amazon EC2, Azure Virtual Machines y Google Compute Engine— distribuían infraestructura a petición según un método de aprovisionamiento de recursos programático y de autoservicio. De este modo, las empresas podían empezar con soluciones de pequeña escala y ampliarlas con rapidez cuando fuese necesario.

En 2013, Docker, Inc. presentó Docker, una plataforma ligera en el nivel del sistema operativo basada en contenedores Linux. Docker aprovechaba la capacidad inherente de Linux de ejecutar varias aplicaciones aisladas dentro de un mismo sistema operativo.

Tradicionalmente, la virtualización permitía ejecutar varios sistemas operativos dentro de uno solo. Los hipervisores —componentes de software que gestionan la virtualización de los recursos de computación— se ocupan de crear particiones de recursos y de aislar rigurosamente las máquinas virtuales. Los contenedores de Linux ofrecen virtualización a nivel del sistema

operativo para ejecutar varios procesos de Linux aislados (contenedores) en un host utilizando un solo kernel de Linux. El kernel de Linux utiliza los grupos de control (o *cgroups*) para supervisar los recursos y los espacios de nombres de Linux para aislar los procesos, y gestiona la priorización de recursos —unidad central de procesamiento (CPU, por sus siglas en inglés), memoria, E/S de bloques, red, etc.—; todo ello sin necesidad de utilizar máquinas virtuales.

A diferencia de las máquinas virtuales, los contenedores no dependen de un hipervisor, sino que comparten los servicios del sistema operativo subyacente en el nivel del kernel. Los contenedores Linux tienen un tamaño más pequeño, se inician a la velocidad normal de un inicio de procesos, se amplían con rapidez y, lo más importante, son portátiles. Un contenedor construido en Ubuntu se puede implementar con rapidez en Red Hat sin hacer ni un solo cambio. Los administradores pueden iniciar las aplicaciones basadas en contenedores en cuestión de milésimas de segundo y llevarlas a cientos de instancias al instante.

Aunque los contenedores ya formaban parte de los sistemas operativos modernos basados en Unix —como Linux (LXC), FreeBSD (Jails) y Solaris (Zones)—, fue Docker, Inc. quien puso la tecnología al alcance de los desarrolladores. Fue toda una revolución para la implementación y el desarrollo de aplicaciones.

Para las organizaciones, la combinación de IaaS y contenedores en la nube pública era una promesa de agilidad y escala sin precedentes. La posibilidad de lanzar decenas —en ciertos casos, incluso cientos— de contenedores en cada máquina virtual permitía aprovechar al máximo los recursos de la CPU y la memoria de cada máquina virtual (MV). Los contenedores implementados en la nube pública permitían ejecutar cargas de trabajo a escala web a un coste asequible. La potente combinación de IaaS y contenedores se convirtió en el ingrediente secreto de las empresas emergentes a escala web.

Aunque Docker, Inc. facilitaba las herramientas y el tiempo de ejecución del contenedor necesarios para gestionar todo el ciclo de vida de un contenedor, los profesionales del sector se dieron cuenta de que hacía falta una plataforma para gestionar cientos de contenedores que se ejecutaban en cientos de máquinas virtuales. Esta necesidad desembocó en el lanzamiento de Docker Swarm por parte de Docker, Inc. y de DC/OS por parte de D2iQ (antes denominado Mesosphere).

Mucho antes de que los contenedores se ganasen la estima de los desarrolladores, Google ya ejecutaba algunos de sus principales servicios web en contenedores Linux. En una ponencia presentada en GlueCon 2014, Joe Beda (uno de los creadores de Kubernetes) aseguró que Google lanzaba más de dos mil millones de contenedores a la semana. El secreto de Google para conseguir gestionar esa ingente cantidad de contenedores era su herramienta interna de gestión del centro de datos, Borg.

En junio de 2014, Google lanzó una plataforma de software de código abierto para gestionar contenedores a gran escala, llamada Kubernetes. Era una versión de Borg más imparcial. Google incorporó a Kubernetes lo mejor de Borg y resolvió los puntos débiles que habían señalado los usuarios a lo largo de los años.

En 2015, Kubernetes 1.0 pasó a manos de The Linux Foundation, que después creó la Cloud Native Computing Foundation (CNCF) para gestionar y gobernar el proyecto. En la actualidad, la CNCF custodia varios proyectos de código abierto relacionados con los contenedores, como Containerd, Envoy y Prometheus, entre otros.

¿Dónde encajaba Docker en todo esto? Su propósito era simplificar la creación de aplicaciones modernas. El desarrollador instalaba Docker Engine en su estación de trabajo y, a continuación, utilizaba las herramientas e interfaces de programación de aplicaciones (API, por sus siglas en inglés) de Docker para gestionar el ciclo de vida de las aplicaciones basadas en contenedores. Docker creó Compose y Swarm como extensión de Docker Engine, lo que permitía utilizar el flujo de trabajo y la cadena de herramientas ya conocidos para implementar y gestionar cargas de trabajo con varios contenedores en distintos nodos o máquinas.

Pero fue Google quien dio un paso más al posibilitar la ejecución de distintos tipos de cargas de trabajo basadas en contenedores a gran escala en Kubernetes. Gracias a su extensibilidad, escala y variedad de entornos de implementación posibles, Kubernetes no tardó en convertirse en el favorito de los desarrolladores y operadores.

Apache Mesos, un proyecto de código abierto desarrollado en la Universidad de California en Berkeley, fue una de las arquitecturas de computación distribuida originales para gestionar cargas de trabajo de aplicaciones en clústeres de computación. Pero acabó perdiendo fuerza, ya que el

sector y la comunidad partidaria de los proyectos de código abierto apoyaron el ecosistema de Kubernetes. Mesos es un entorno de computación distribuida ya consolidado, pero resulta más adecuado para sistemas de gran escala con cientos de nodos. En su origen se diseñó para aplicaciones distribuidas basadas en Apache Hadoop, Apache Spark y Kafka, y la orquestación de contenedores llegó mucho más tarde, con el complemento Marathon. En cambio, Kubernetes se había diseñado pensando en los contenedores y se podía ejecutar en clústeres de todos los tamaños con la misma facilidad y flexibilidad.

Docker, Inc. también ha adoptado Kubernetes como herramienta de orquestación de contenedores preferida, integrándolo en Docker Desktop y Docker Enterprise. En noviembre de 2019, Mirantis, una empresa de infraestructuras basadas en OpenStack, adquirió la oferta empresarial de Docker, incluido Kubernetes como servicio.

Por su sencillez, accesibilidad y escalabilidad, Kubernetes acabó convirtiéndose en la plataforma de gestión de contenedores más popular. De hecho, es uno de los proyectos de código abierto que más rápido ha crecido en la historia de la informática. Las aplicaciones modernas y emergentes utilizan Kubernetes cada vez más, lo que ha llevado al auge de las plataformas de gestión de contenedores empresariales como Google Anthos, Red Hat OpenShift y VMware Tanzu. También ha habido un aumento en la oferta de soluciones gestionadas de contenedores como servicio en la nube pública, como Amazon Elastic Kubernetes Service, Azure Kubernetes Service y Google Kubernetes Engine.

El tamaño de la implementación importa

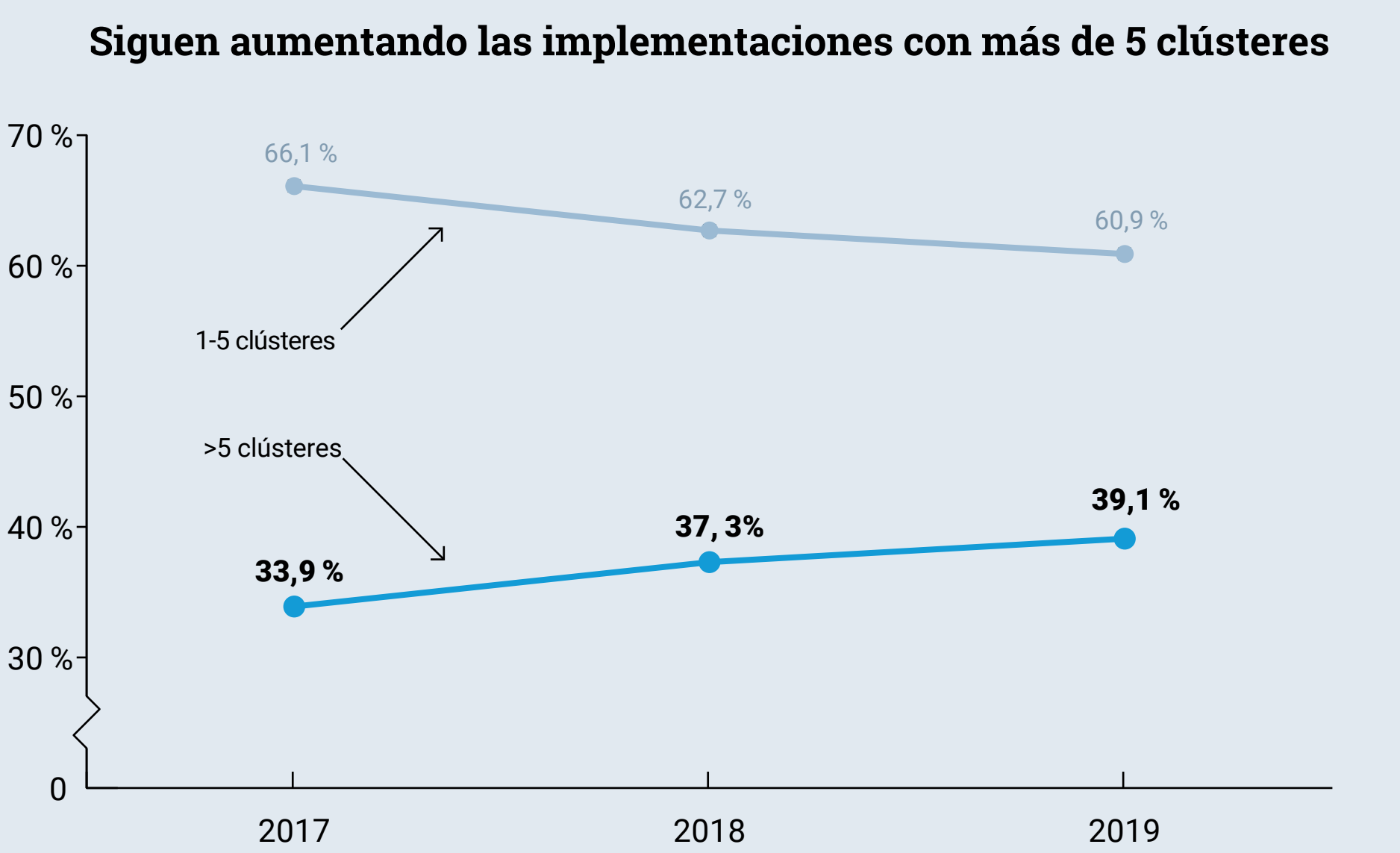
Aunque ya se han resuelto las primeras dificultades que planteaban los contenedores, como la gestión del almacenamiento, los usuarios de Kubernetes siguen afrontando problemas relacionados con la seguridad y con el cambio cultural que implica la adopción de estos sistemas. Además, tienen que adaptarse a las novedades que se van introduciendo, como el uso de una malla de servicios, que suele recurrir a proxies sidecar para controlar los microservicios dentro de los entornos de Kubernetes. El objetivo de la malla de servicios es mejorar el control del tráfico de las aplicaciones, la observabilidad y la seguridad de los sistemas distribuidos. Veamos qué

dicen los datos de la encuesta anual de la CNCF.

Las cifras dejan claro el camino recorrido desde las primeras pruebas hasta las implementaciones importantes de Kubernetes. En 2017, el 64 % de los más de 550 encuestados en el estudio anual de la CNCF utilizaban el proyecto Kubernetes en el entorno de producción. En 2018, el interés por Kubernetes había aumentado y más de 2400 personas encuestadas ya lo estaban evaluando. En 2019, el 78 % de los más de 1300 encuestados utilizaban Kubernetes en el entorno de producción, lo cual confirmaba una vez más el lugar destacado que ocupaba Kubernetes en la comunidad nativa en la nube de la CNCF.

¿Qué fue antes, el huevo o la gallina? En un principio, la mayor adopción de contenedores generó una demanda de orquestadores de contenedores, pero la implementación de la infraestructura de Kubernetes facilita la ejecución de una mayor cantidad de ellos. Por lo general, los usuarios de Kubernetes ejecutan más contenedores, pero la forma de reaccionar a su proliferación no ha sido uniforme.

FIG. 1.1: *Predominan las implementaciones de mayor tamaño. El porcentaje de usuarios de Kubernetes con más de cinco clústeres pasó del 34 % en 2017 al 39 % en 2019.*



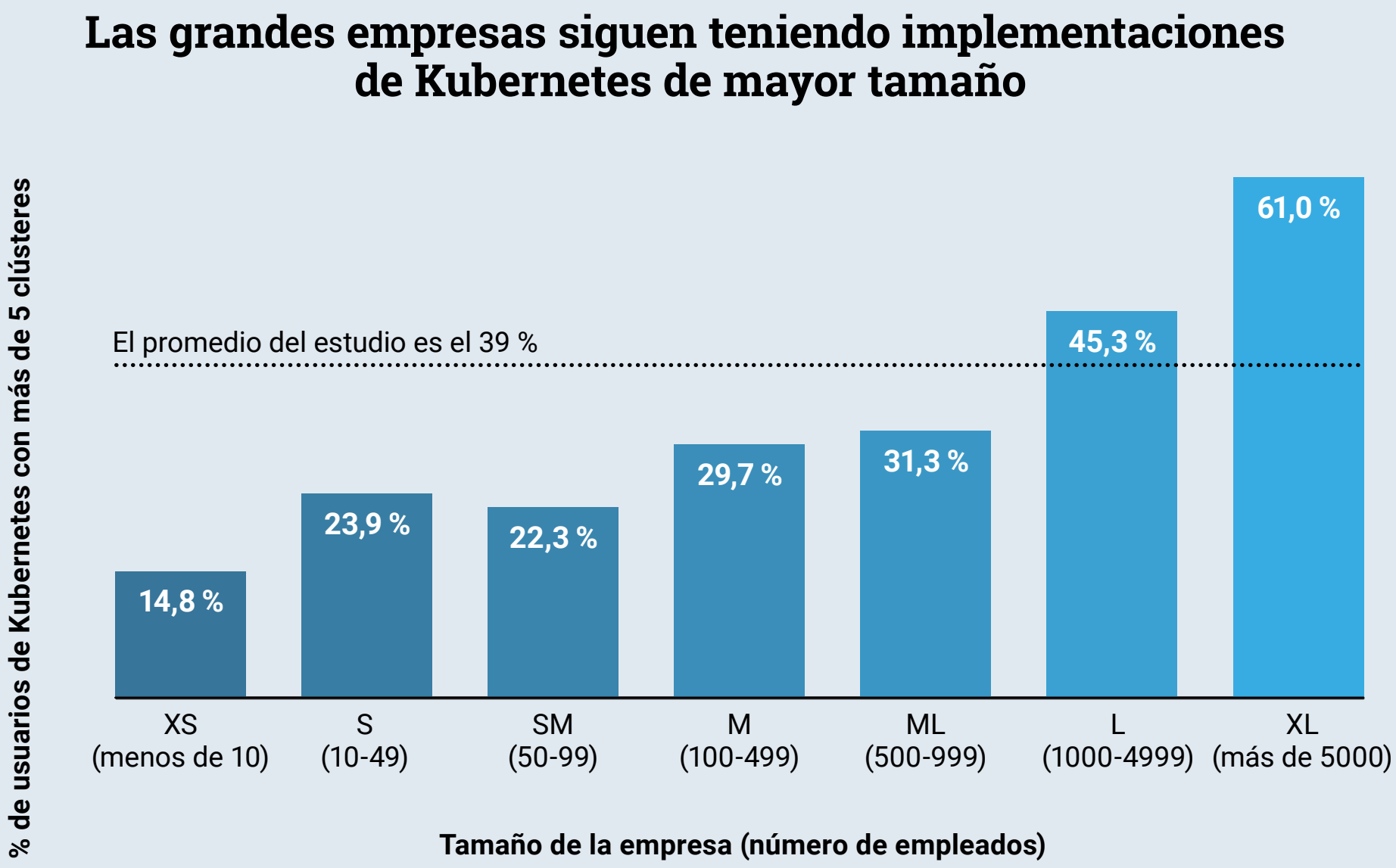
Fuente: Análisis de The New Stack de las encuestas de la CNCF realizadas en 2017, 2018 y 2019.
P: ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan. Los datos solo se refieren a los encuestados que tienen al menos un clúster de Kubernetes.
2017, n=454; 2018, n=1475; 2019, n=1197.

Las implementaciones de Kubernetes con más de cinco clústeres no dejan de aumentar: como se muestra a continuación en la figura 1.1, han pasado del 34 % en 2017 al 39 % en 2019. De hecho, más de la mitad de este grupo ejecuta mil contenedores o más (en 2017 lo hacía el 55 %, y en 2019 el porcentaje ya era el 62 %).

Podemos concluir lo siguiente:

- La adopción se concentra entre las grandes empresas, que tienden a quedarse con sus proveedores ya existentes para la implementación de Kubernetes.
- El uso de contenedores ha aumentado, pero la adaptación resulta más difícil para las organizaciones con un máximo de cinco clústeres de Kubernetes.
- Se están utilizando diversas soluciones de almacenamiento, entrada y malla de servicios, tanto emergentes como ya consolidadas. A la hora de responder a las dificultades, es más probable que se tengan en cuenta las herramientas emergentes, pero, al mismo tiempo,

FIG. 1.2: *En las organizaciones con un mínimo de 5000 empleados, el 61 % de los usuarios de Kubernetes tienen más de cinco clústeres. El promedio del estudio es el 39 %.*



Fuente: Análisis de The New Stack de la encuesta de la CNCF realizada en 2019.
P: Si utiliza Kubernetes, ¿cuántos clústeres tiene en el entorno de producción?
Menos de 10, n=81; 10-49, n=142; 50-99, n=121; 100-499, n=239; 500-999, n=80; 1000-4999, n=170; 5000 o más, n=364.

los clientes tienden a mantener las relaciones con los proveedores que ya utilizan.

La figura 1.2 muestra que el número de empleados, y no tanto la cantidad de contenedores, tiene una relación directa con el tamaño de las implementaciones de Kubernetes. En las organizaciones con un mínimo de 5000 empleados, el 61 % de los usuarios de Kubernetes tienen más de cinco clústeres.

Si ampliamos un poco la muestra, entre las organizaciones con al menos mil empleados, es más probable la presencia de más de cinco clústeres (el porcentaje pasó del 51 % en 2017 al 56 % en 2019).

Estas cifras son significativas, porque las empresas con implementaciones de Kubernetes de gran tamaño han logrado reducir mucho más los problemas relacionados con los contenedores.

En las organizaciones con un máximo de cinco clústeres, no resulta fácil gestionar una gran cantidad de contenedores.

¿Qué significa «orquestación»?

La infraestructura moderna se ha visto influida por dos tendencias: los contenedores y la metodología DevOps. El ecosistema de DevOps ha evolucionado para ofrecer integración continua, pruebas continuas, implementación continua y supervisión continua, lo cual ha acelerado el desarrollo de software. La adopción de contenedores en combinación con las prácticas recomendadas de DevOps de eficacia probada ha llevado a una mayor rapidez de las implementaciones a gran escala.

Mientras los contenedores contribuyen a mejorar la productividad de los desarrolladores, las herramientas de orquestación brindan numerosas ventajas a las organizaciones que tratan de optimizar sus inversiones en operaciones y DevOps. Entre dichas ventajas, se encuentran las siguientes:

- Eficiencia en la gestión de recursos.
- Fluidez a la hora de ampliar la escala de los servicios.

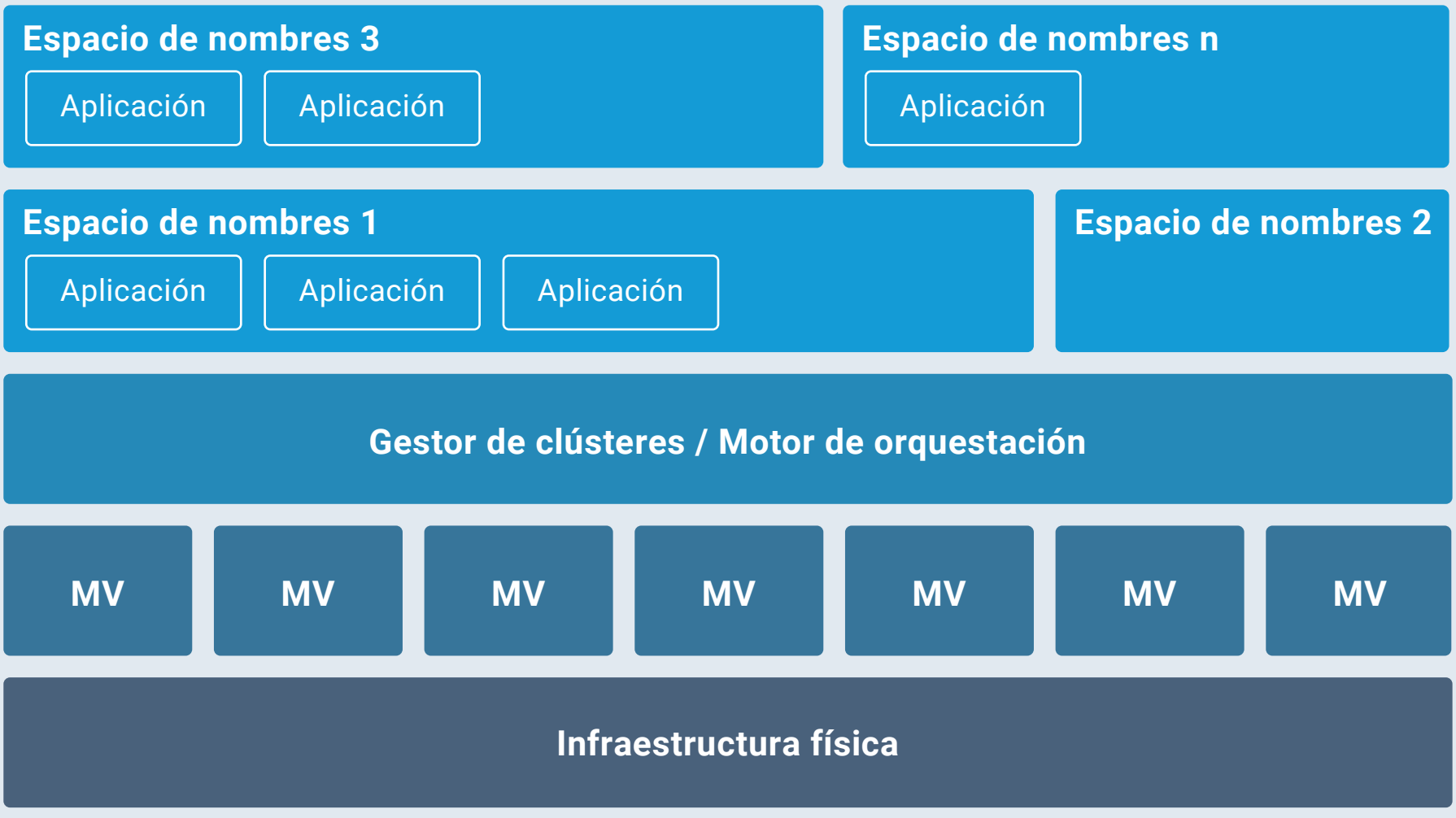


FIG. 1.3: *Las capas de recursos de un sistema, desde el punto de vista del motor de orquestación de contenedores.*

- Alta disponibilidad.
- Ahorro de costes operativos a gran escala.
- Modelo declarativo para la mayoría de las herramientas de orquestación, lo que reduce las fricciones y permite una gestión más autónoma.
- Infraestructura como servicio (IaaS) que recuerda a las operaciones pero se gestiona como una plataforma como servicio (PaaS, por sus siglas en inglés).
- Experiencia operativa homogénea entre los proveedores de nube pública y de soluciones locales.

Los operadores eligen la infraestructura IaaS para el control y la automatización, mientras que los desarrolladores prefieren la plataforma PaaS por su flexibilidad, escala y productividad. Las herramientas de orquestación de contenedores combinan las ventajas de ambas: automatización y escala.

Los contenedores han resuelto el problema de la productividad de los desarrolladores y han garantizado la fluidez del flujo de trabajo de DevOps gracias a la posibilidad de crear una imagen

de un contenedor, ejecutar un contenedor y desarrollar código en dicho contenedor para implementarlo en centros de datos locales o entornos de nube pública. Sin embargo, esta fluidez en la productividad de los desarrolladores no se traduce automáticamente en una mayor eficiencia en los entornos de producción.

El entorno de producción suele ser bastante diferente del entorno local que tiene un desarrollador en su portátil. Independientemente de que se ejecuten aplicaciones tradicionales de tres niveles a gran escala o aplicaciones basadas en microservicios, no resulta nada fácil gestionar una gran cantidad de contenedores y el clúster de nodos en que se basan. La orquestación es el componente necesario para ampliar la escala, pues toda implementación a gran escala requiere automatización.

El carácter distribuido de la computación en la nube trajo consigo un cambio de paradigma en el modo de percibir la infraestructura de máquinas virtuales. La analogía «pet vs. cattle» (mascotas frente a ganado) —tratar a los contenedores como si fueran cabezas de ganado reemplazables, a diferencia de los animales de compañía— favoreció un cambio de mentalidad en cuanto a la naturaleza de los contenedores y la infraestructura.

Tanto los contenedores como la infraestructura en la que se ejecutan son inmutables, es decir, los contenedores o servidores no se modifican nunca una vez implementados. Si hay que actualizar, arreglar o modificar algo, se sustituyen los antiguos contenedores o servidores por otros nuevos creados a partir de una imagen común con los cambios necesarios. Este sistema se puede comparar al modo de gestionar el ganado en una explotación lechera.

En cambio, los servidores tradicionales e incluso las máquinas virtuales no se consideran inmutables, de ahí que en la analogía se los compare con los animales de compañía (no son desechables). En consecuencia, su coste de mantenimiento es elevado, porque requieren la atención constante del equipo de operaciones.

La infraestructura inmutable es programable, lo cual hace posible la automatización. La infraestructura como código (IaC, por sus siglas en inglés) es uno de los atributos clave de la infraestructura moderna, que permite que una aplicación aprovisione, configure y aproveche la infraestructura de forma programática para ejecutarse.

La combinación de orquestación de contenedores, infraestructura inmutable y automatización basada en IaC garantiza la flexibilidad y la escalabilidad.

En la práctica, el uso de contenedores a gran escala amplía y perfecciona los conceptos de escalabilidad y disponibilidad de los recursos.

Las plataformas de orquestación de contenedores incluyen funciones de referencia como las siguientes:

- programación;
- gestión de recursos;
- detección de servicios;
- comprobación del estado de salud;
- escalabilidad automática;
- actualizaciones y mejoras.

En la actualidad, el mercado de la orquestación de contenedores está dominado por Kubernetes, que se ha ganado la aprobación de las grandes empresas, los proveedores de plataformas y de servicios en la nube y las empresas de infraestructuras.

La orquestación de contenedores fomenta el uso de las arquitecturas de microservicios, donde cada aplicación está formada por servicios independientes de menor tamaño, cada uno de ellos diseñado para una tarea en concreto. Cada microservicio se empaqueta como un contenedor y Kubernetes orquesta en el tiempo de ejecución el conjunto de microservicios pertenecientes lógicamente a una misma aplicación.

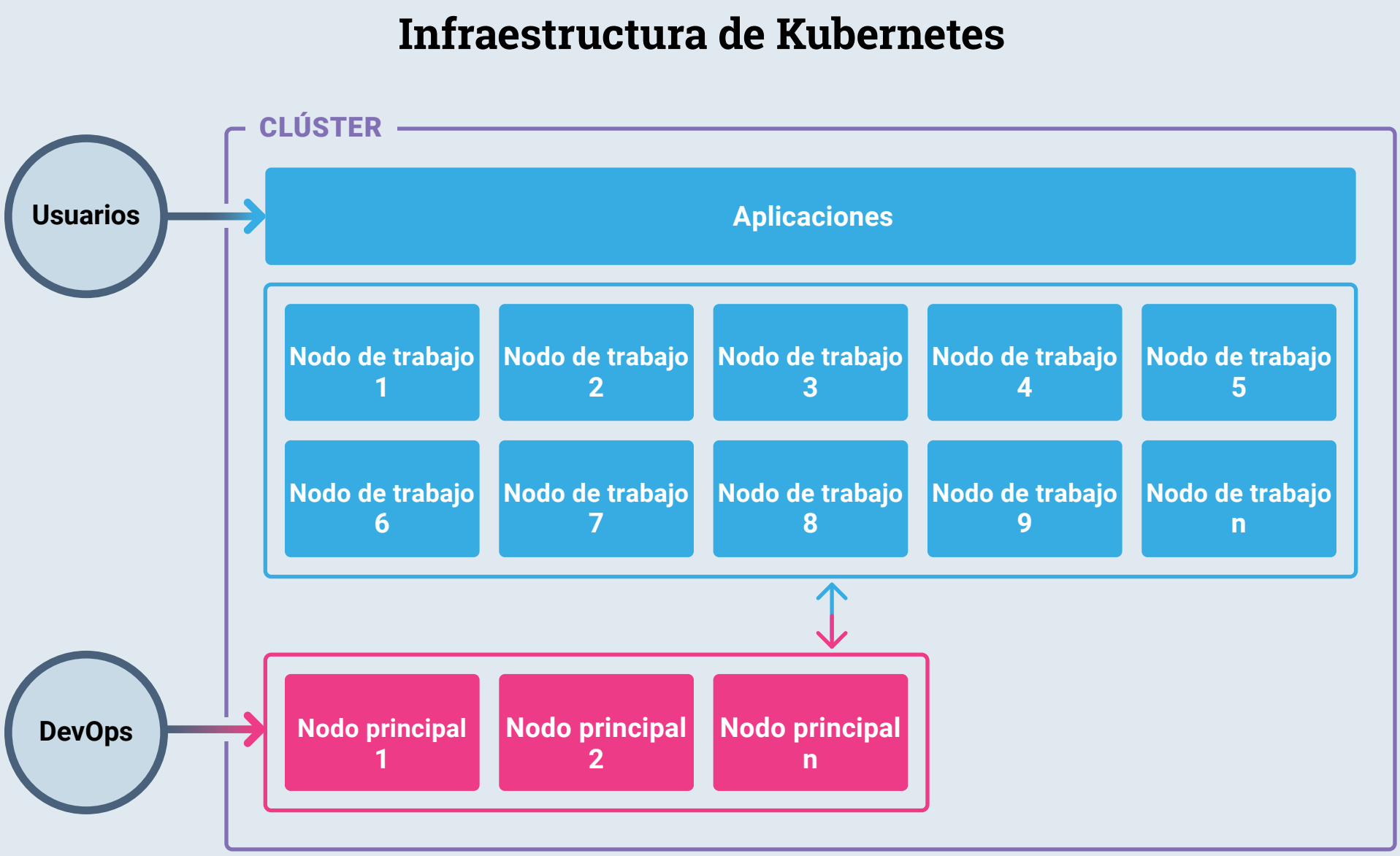
El auge de Kubernetes ha generado nuevos segmentos de mercado basados en la orquestación de contenedores y las plataformas de gestión. En los ámbitos más diversos (el almacenamiento, las redes, la supervisión o la seguridad), hay una nueva generación de empresas y empresas emergentes que están creando productos y servicios nativos para contenedores. En el siguiente capítulo, destacamos algunos de los componentes básicos de las plataformas nativas en la nube y varias empresas emergentes de este nuevo ecosistema.

Arquitectura de Kubernetes

Una aplicación contemporánea, empaquetada en forma de conjunto de contenedores e implementada como microservicios necesita una infraestructura lo suficientemente sólida como para responder a las exigencias de los clústeres y a la presión que supone la orquestación dinámica. Una infraestructura de este tipo debería proporcionar primitivas (o bloques de construcción) para programar, supervisar, actualizar y reubicar contenedores en los distintos hosts, y debería tratar las primitivas subyacentes de computación, almacenamiento y redes como una reserva de recursos. Cada carga de trabajo basada en contenedores debería poder aprovechar los recursos expuestos a ella, como los núcleos de CPU, las unidades de almacenamiento y las redes.

Kubernetes (véase la figura 1.4) es un sistema distribuido de código abierto que abstrae la infraestructura física subyacente, con lo que facilita la ejecución a gran escala de aplicaciones basadas en contenedores. Una aplicación gestionada por Kubernetes durante todo su ciclo de vida

FIG. 1.4: *Visión de conjunto de un clúster de Kubernetes.*

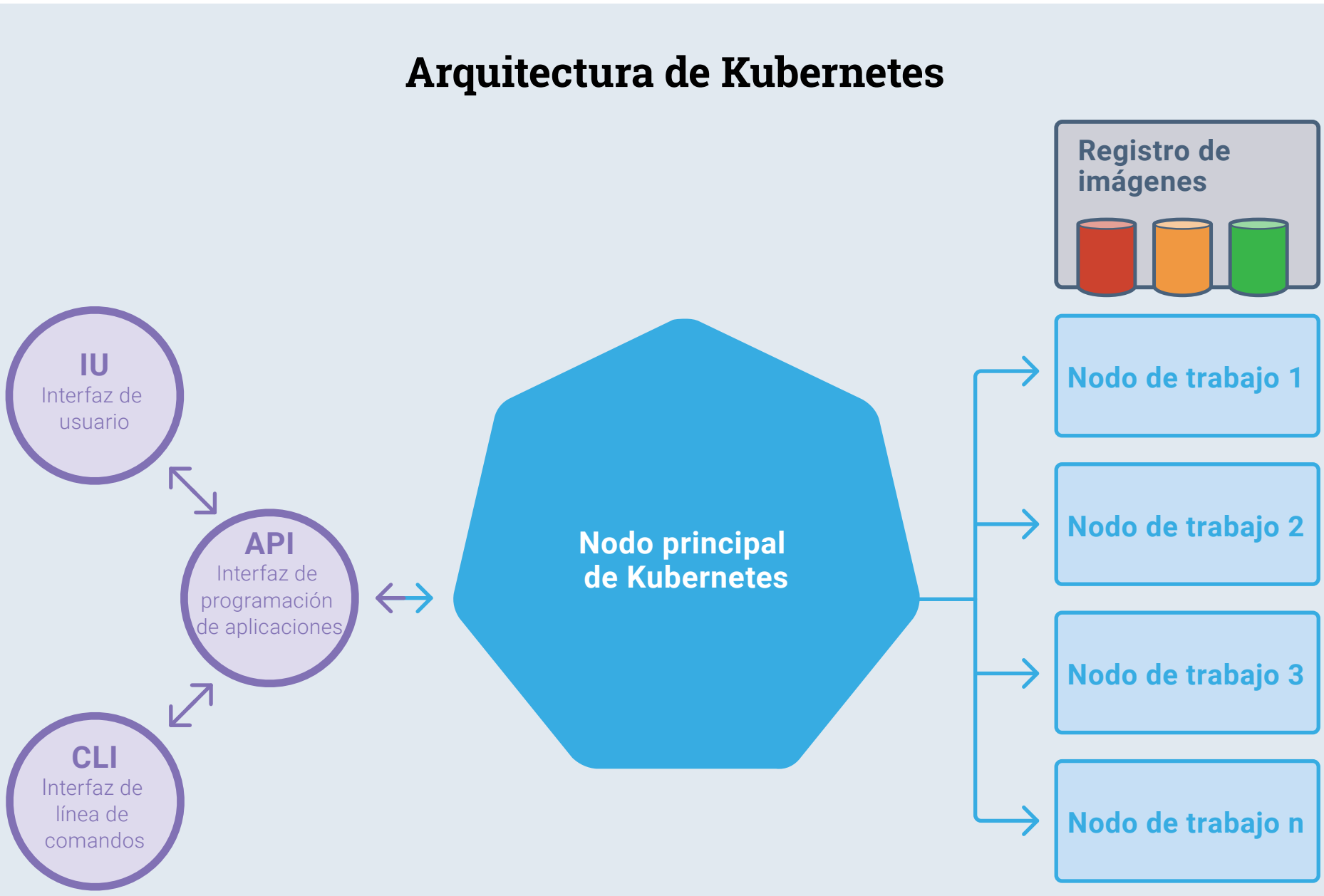


Fuente: Janakiram MSV

está formada por contenedores reunidos en un conjunto y coordinados en una sola unidad. Una capa de gestión de clústeres que sea eficiente permite a Kubernetes desvincular la aplicación de su infraestructura subyacente, tal como se muestra en la siguiente figura. Cuando la infraestructura de Kubernetes está completamente configurada, los equipos de DevOps pueden centrarse en gestionar las cargas de trabajo implementadas en lugar de ocuparse de la reserva de recursos subyacente —CPU y memoria—, pues de eso se encarga Kubernetes.

Kubernetes es un ejemplo de sistema distribuido bien construido. Trata todas las máquinas contenidas en un clúster como una sola reserva de recursos. Desempeña el papel de sistema operativo distribuido mediante la gestión eficaz de la programación, la asignación de recursos, la supervisión del estado de salud de la infraestructura e incluso el mantenimiento del estado deseado de la infraestructura y las cargas de trabajo. Kubernetes es un sistema operativo capaz de ejecutar aplicaciones modernas de distintos clústeres e infraestructuras en servicios en la nube y entornos de centros de datos privados.

FIG. 1.5: *La función de un nodo principal en la arquitectura de Kubernetes.*



Fuente: Janakiram MSV

Como cualquier otro sistema distribuido consolidado, Kubernetes tiene dos capas: los nodos principales y los nodos de trabajo (véase la figura 1.5). Los nodos principales suelen ejecutar el plano de control encargado de programar y gestionar el ciclo de vida de las cargas de trabajo. Los nodos de trabajo hacen el trabajo pesado a la hora de ejecutar las aplicaciones. La combinación de nodos principales y nodos de trabajo constituye un clúster.

Los equipos de DevOps que gestionan el clúster se comunican con la API del plano de control mediante la interfaz de línea de comandos (CLI, por sus siglas en inglés) o herramientas de terceros. Los usuarios acceden a las aplicaciones que se ejecutan en los nodos de trabajo. Las aplicaciones están formadas por una o varias imágenes de contenedor que se almacenan en un registro de imágenes accesible.

Plano de control

El plano de control ejecuta los componentes de Kubernetes que proporcionan las funciones clave: exposición de la API de Kubernetes, programación de las implementaciones de cargas de trabajo, gestión del clúster y direccionamiento de las comunicaciones en todo el sistema. Como se ilustra en la figura 1.5, el nodo principal supervisa los contenedores que se ejecutan en cada nodo, así como el estado de todos los nodos registrados. Las imágenes de contenedor, que funcionan como artefactos que se pueden implementar, deben estar disponibles para el clúster de Kubernetes mediante un registro de imágenes público o privado. Los nodos encargados de programar y ejecutar las aplicaciones acceden a las imágenes del registro mediante el tiempo de ejecución del contenedor.

Como se observa en la figura 1.6, el nodo principal de Kubernetes ejecuta los siguientes componentes que constituyen el plano de control:

etcd

etcd, desarrollado por CoreOS (más tarde adquirido por Red Hat), es un almacén de datos clave-valor persistente, ligero y distribuido que mantiene los datos de configuración del clúster. Representa el estado general del clúster en un momento determinado y funciona como única fuente de información fidedigna. Otros componentes y servicios observan si se producen cambios en el almacén etcd para mantener el estado deseado de cada aplicación. Dicho estado se define con

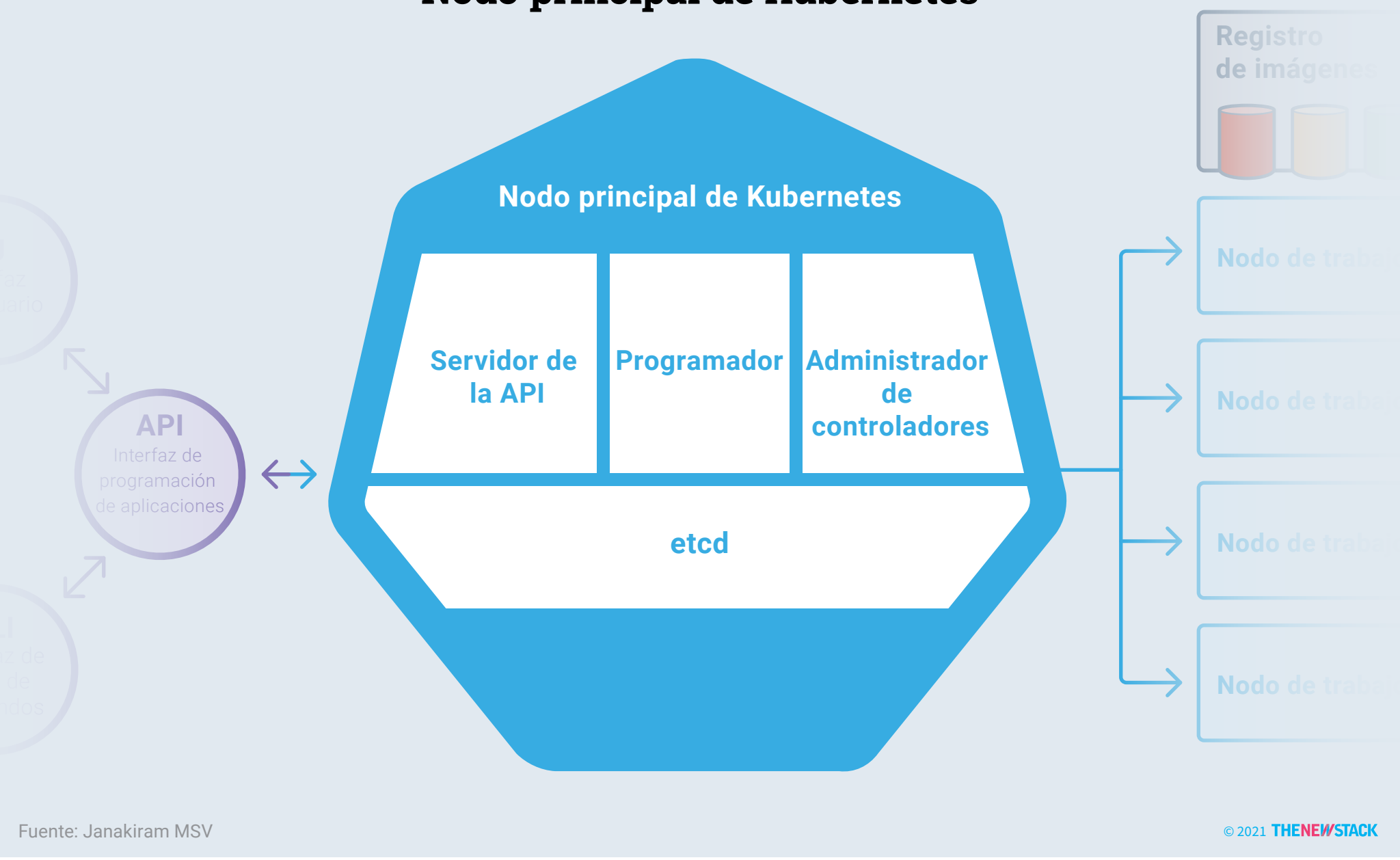


FIG. 1.6: *Componentes del nodo principal.*

una política declarativa (en la práctica, un documento que establece el entorno óptimo para dicha aplicación, de forma que el orquestador pueda intervenir para conseguir tal entorno). Esta política determina el modo en que el orquestador aborda las distintas propiedades de una aplicación, como el número de instancias, los requisitos de almacenamiento y la asignación de recursos.

Solo se puede acceder a la base de datos de etcd mediante el servidor de la API. Todo componente del clúster que tenga que leer o escribir en etcd lo hace mediante el servidor de la API.

Servidor de la API

El servidor de la API expone la API de Kubernetes mediante JSON a través de HTTP, con lo que proporciona la interfaz de transferencia de estado representacional (REST, por sus siglas en inglés) para los endpoints internos y externos del orquestador. La CLI, la interfaz de usuario (IU) web u otra herramienta pueden enviar solicitudes al servidor de la API. El servidor procesa y valida la solicitud y, a continuación, actualiza el estado de los objetos de la API en etcd. De este modo, los clientes pueden configurar las cargas de trabajo y los contenedores en los distintos nodos de trabajo.

Programador

El programador evalúa la disponibilidad de recursos y selecciona en consecuencia el nodo en el que se debe ejecutar cada carga de trabajo. A continuación, hace un seguimiento del uso de los recursos para garantizar que el pod no supere los recursos asignados. Se ocupa del mantenimiento y seguimiento de los requisitos y la disponibilidad de los recursos, así como de otras directivas y limitaciones indicadas por los usuarios; por ejemplo, la calidad del servicio (QoS, por sus siglas en inglés), los requisitos de afinidad/antiafinidad y la localidad de los datos. Un equipo de operaciones puede definir el modelo de recursos de forma declarativa. El programador interpreta dichas declaraciones como instrucciones para aprovisionar y asignar el conjunto de recursos adecuado a cada carga de trabajo.

Administrador de controladores

La parte de la arquitectura de Kubernetes a la que este debe su versatilidad es el administrador de controladores, que forma parte del nodo principal. Se encarga de garantizar que el clúster mantenga el estado deseado de las aplicaciones en todo momento mediante un controlador bien definido. Un controlador es un bucle de control que vigila el estado compartido del clúster mediante el servidor de la API y realiza cambios para intentar acercar el estado actual al estado deseado.

Para mantener la estabilidad de los nodos y pods, el controlador supervisa constantemente el estado del clúster y las cargas de trabajo implementadas en él. Por ejemplo, cuando un nodo presenta algún problema, es posible que los pods que se ejecutan en dicho nodo dejen de estar accesibles. En ese caso, el controlador se encarga de programar el mismo número de pods nuevos en un nodo diferente, lo que garantiza que el clúster mantenga el estado esperado en cualquier momento dado.

Kubernetes contiene una serie de controladores integrados que se ejecutan dentro del administrador de controladores. Estos controladores ofrecen primitivas coordinadas que se ajustan con cierto tipo de cargas de trabajo, como tareas cron sin estado, con estado y programadas, y tareas de ejecución hasta el final. Los desarrolladores y operadores pueden utilizar estas primitivas al empaquetar e implementar aplicaciones en Kubernetes.

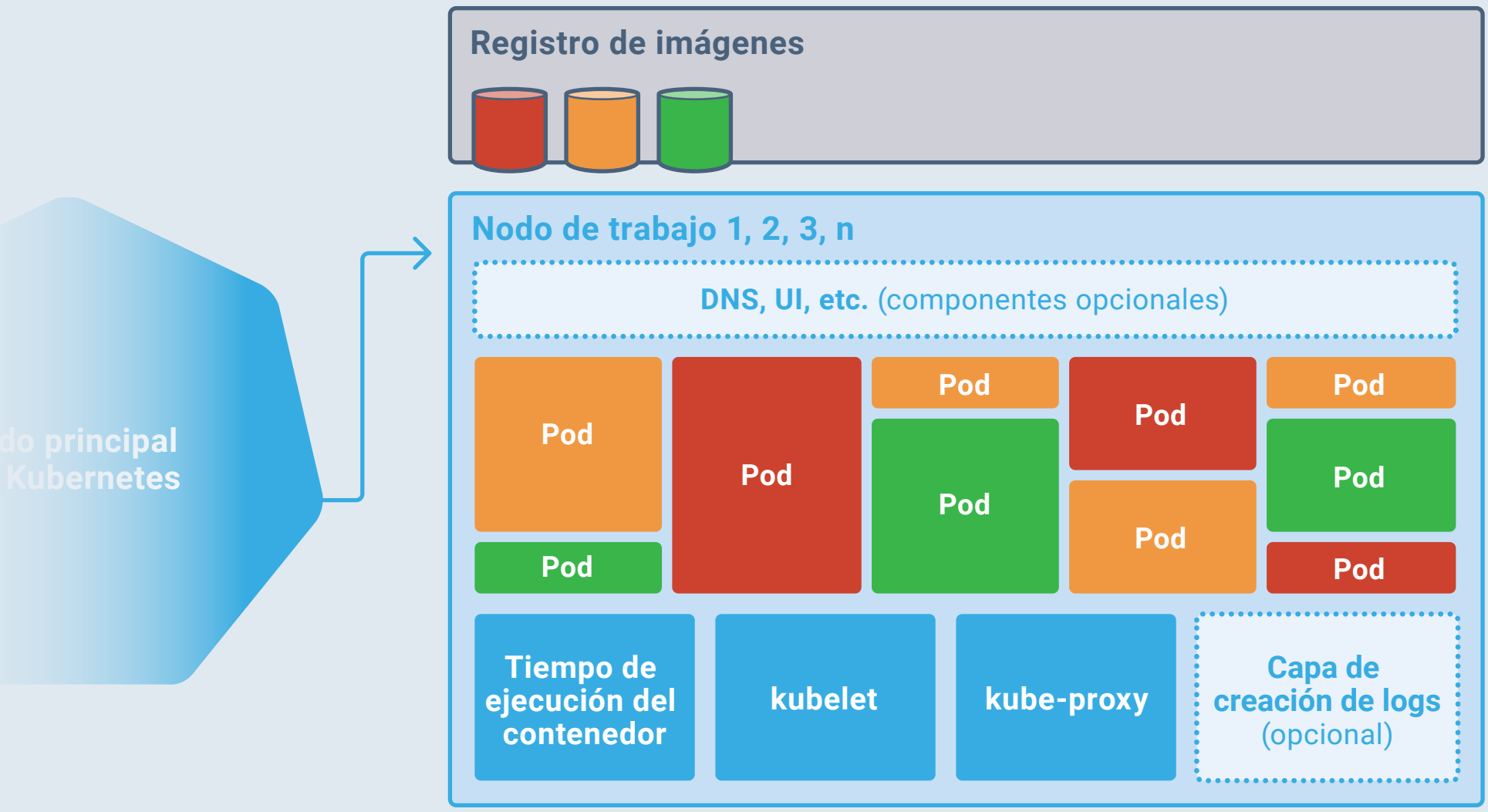


FIG. 1.7: *Componentes de los nodos de trabajo.*

Nodos de trabajo

El nodo de trabajo es el «mulo de carga» del clúster de Kubernetes, encargado de ejecutar las cargas de trabajo basadas en contenedores; componentes adicionales de creación de logs, supervisión y detección de servicios; y complementos opcionales. Su cometido es exponer los recursos de computación, conectividad y almacenamiento a las aplicaciones. Cada nodo contiene un tiempo de ejecución del contenedor, como Docker, y un agente (kubelet) que se comunica con el nodo principal. Un nodo puede ser una máquina virtual (MV) que se ejecuta en una nube o un servidor físico dentro de un centro de datos.

Como se muestra en la figura 1.7, cada nodo contiene los siguientes componentes:

Tiempo de ejecución del contenedor

El tiempo de ejecución del contenedor se encarga de gestionar el ciclo de vida de cada contenedor que se ejecuta en el nodo. Una vez programado un pod en el nodo, el tiempo de ejecución extrae del registro las imágenes especificadas por el pod. Cuando se finaliza un pod, el tiempo de

ejecución elimina los contenedores pertenecientes a dicho pod. Kubernetes puede comunicarse con cualquier tiempo de ejecución del contenedor compatible con la Open Container Initiative (OCI), como Docker y CRI-O.

La OCI es un estándar que define la especificación del tiempo de ejecución y de las imágenes, con el objetivo de impulsar la estandarización de los tiempos de ejecución de los contenedores y los formatos de las imágenes.

Kubelet

El kubelet es el agente de Kubernetes encargado de interactuar con el tiempo de ejecución del contenedor para realizar operaciones como el inicio, la detención y el mantenimiento de contenedores.

Además, cada kubelet supervisa el estado de los pods. Cuando un pod no cumple el estado deseado indicado en la implementación, se puede reiniciar en el mismo nodo. El estado del nodo se transmite al nodo principal cada pocos segundos mediante mensajes de latido. Si el nodo principal detecta un fallo en el nodo, el controlador de replicación observa este cambio de estado y programa los pods en otros nodos que no presenten problemas.

Kube-proxy

El componente `kube-proxy` se implementa como proxy de red y equilibrador de carga que orquesta la red para dirigir las solicitudes a los pods adecuados. Así, dirige el tráfico al pod que corresponda según el nombre de servicio asociado y el número de puerto de la solicitud entrante. Además, aprovecha las funciones de red específicas del sistema operativo mediante la manipulación de las políticas y reglas definidas con `iptables`. Cada componente `kube-proxy` se puede integrar con capas de red como Calico y Flannel.

Capa de creación de logs

El orquestador utiliza con frecuencia la creación de logs como forma de recopilar parámetros sobre el rendimiento y el consumo de recursos relativos a los contenedores de cada nodo, como el uso de la CPU, la memoria, los archivos y las redes. La Cloud Native Computing Foundation aloja un componente de software llamado Fluentd, que proporciona una capa de creación de logs unificada para su uso con Kubernetes u otros orquestadores. Este componente genera parámetros

que el controlador principal de Kubernetes necesita para llevar un seguimiento de los recursos del clúster disponibles, así como del estado de la infraestructura en general.

Complementos

Kubernetes admite servicios adicionales en forma de complementos. Estos servicios opcionales, como el panel, se implementan igual que otras aplicaciones, pero se integran con otros componentes clave del nodo, como la capa de creación de logs y `kube-proxy`. Por ejemplo, el complemento del panel extrae parámetros del kubelet para mostrar vistas detalladas del consumo de recursos. El complemento DNS, que se basa en `kube-dns` o CoreDNS, optimiza `kube-proxy` mediante la resolución de nombres.

Cargas de trabajo

Así como el plano de control y los nodos de trabajo constituyen la infraestructura del clúster principal, las cargas de trabajo son las aplicaciones basadas en contenedores implementadas en Kubernetes.

Después de desarrollar y probar un microservicio, los desarrolladores lo empaquetan en forma de contenedor, que es la unidad más pequeña de implementación empaquetada como pod. Un conjunto de contenedores pertenecientes a la misma aplicación se agrupa, empaqueta, implementa y gestiona dentro de Kubernetes.

Kubernetes expone las primitivas para la implementación, al tiempo que amplía, detecta y supervisa constantemente el estado de estos microservicios. Se suelen utilizar espacios de nombres para separar de forma lógica una aplicación de otra. Funcionan como un clúster lógico que determina un alcance y un límite bien definidos para todos los recursos y servicios pertenecientes a una aplicación.

Dentro de un espacio de nombres, se implementan las siguientes primitivas de Kubernetes:

Pods

Un pod es la unidad de ejecución básica de una aplicación de Kubernetes. Constituye la unidad más pequeña y más sencilla del modelo de objetos de Kubernetes. Un pod también es el elemento programable más pequeño de una aplicación de Kubernetes. Si Kubernetes es un sistema

operativo, el pod constituye un conjunto de procesos —cada uno de los cuales se puede asignar a un contenedor— que se ejecutan en el clúster.

El pod funciona como unidad central de gestión de cargas de trabajo para Kubernetes y representa el límite lógico para los contenedores que comparten los mismos recursos y contexto de ejecución. Al agrupar los contenedores en pods, se posibilita la ejecución simultánea de varios procesos dependientes, con lo que se compensan las dificultades de configuración que surgieron cuando los contenedores sustituyeron a la virtualización de primera generación.

Cada pod reúne uno o varios contenedores que utilizan la comunicación entre procesos (IPC, por sus siglas en inglés) y que pueden compartir las soluciones de almacenamiento y de conectividad. Cuando los contenedores tengan que estar vinculados y ubicados en un mismo lugar —por ejemplo, un contenedor de servidor web y un contenedor de caché—, resulta sencillo empaquetarlos en un solo pod. La escala horizontal de un pod se puede ampliar de forma manual o bien mediante una política definida con la función llamada Horizontal Pod Autoscaling (HPA). Con este método, el número de pods que forman parte de la implementación aumenta de manera proporcional según los recursos disponibles.

Los pods hacen posible una separación funcional entre el desarrollo y la implementación. Mientras los desarrolladores se concentran en su código, los operadores pueden tener una visión más amplia para decidir qué contenedores relacionados entre sí se pueden unir en una unidad funcional. El resultado es la cantidad óptima de portabilidad, pues un pod no es más que una colección de una o varias imágenes de contenedor que se gestionan juntas.

Controladores

En Kubernetes, los controladores proporcionan a los pods funciones adicionales, como características del tiempo de ejecución y la configuración deseada.

Una implementación aporta actualizaciones declarativas a los pods. Para garantizar que se mantenga siempre el estado deseado, lleva un seguimiento del estado de salud de los pods que participan en la implementación. Cada implementación gestiona un ReplicaSet, que mantiene un conjunto estable de pods de réplica ejecutándose en un momento determinado, en función del estado deseado.

Las implementaciones dan a los pods capacidades similares a las de las PaaS mediante la escalabilidad, el historial de implementación y las funciones de restauración. Cuando se configura una implementación con un mínimo de dos réplicas, Kubernetes garantiza que al menos dos pods estén siempre en ejecución, con lo que se obtiene tolerancia a los fallos. Aunque se implemente el pod con una sola réplica, es muy recomendable utilizar un controlador de implementación en lugar de una especificación de pod normal.

Un StatefulSet es similar a una implementación, pero está indicado para los pods que necesiten persistencia, un identificador bien definido y un orden de creación garantizado. Para las cargas de trabajo como los clústeres de bases de datos, un controlador del StatefulSet creará un conjunto de pods con una alta disponibilidad en un orden determinado y con una nomenclatura predecible. Las cargas de trabajo con estado que necesiten una alta disponibilidad, como Cassandra, Kafka, SQL Server y ZooKeeper, se implementan en Kubernetes como StatefulSets.

Para obligar a un pod a ejecutarse en cada nodo del clúster, se puede utilizar un controlador del DaemonSet. Como Kubernetes programa automáticamente un DaemonSet en los nodos de trabajo recién aprovisionados, se convierte en un candidato ideal para configurar y preparar los nodos para la carga de trabajo. Por ejemplo, si hay que montar en los nodos un sistema de archivos compartidos Gluster o un sistema de archivos de red (NFS, por sus siglas en inglés) existente antes de implementar la carga de trabajo, se recomienda empaquetar e implementar el pod en forma de DaemonSet. Los agentes de supervisión son buenos candidatos a ser utilizados como DaemonSet, para garantizar que cada nodo ejecute el agente de supervisión.

Para el procesamiento por lotes y la programación de tareas, los pods se pueden empaquetar para una tarea cron o de ejecución hasta el final. Una tarea crea uno o varios pods y garantiza que se terminen una determinada cantidad de ellos con éxito. Los pods configurados para ejecutarse hasta el final ejecutan la tarea y salen, mientras que las tareas cron ejecutan una tarea según la programación definida en el formato crontab.

Los controladores definen el ciclo de vida de los pods según las características de la carga de trabajo y el contexto de ejecución.

Servicios y detección de servicios

El modelo de servicios de Kubernetes proporciona el aspecto de los microservicios más básico pero también más importante: la detección.

En Kubernetes, cualquier objeto de la API, como un nodo o un pod, puede llevar asociados pares clave-valor, metadatos adicionales para identificar y agrupar objetos que compartan un mismo atributo o propiedad. Kubernetes hace referencia a estos pares clave-valor en forma de etiquetas y anotaciones. La detección de servicios recurre a las etiquetas y a los selectores para asociar un servicio con un conjunto de pods.

Un solo pod o ReplicaSet se puede exponer a clientes internos o externos mediante servicios, que asocian un conjunto de pods con un criterio específico. Cualquier pod cuyas etiquetas coincidan con el selector definido en el manifiesto de servicio será detectado automáticamente por el servicio. Esta arquitectura brinda un mecanismo flexible, sin dependencia fija, para la detección de servicios.

Cuando se crea un pod, se le asigna una dirección IP a la que solo se puede acceder desde dentro del clúster. Pero nada garantiza que la dirección IP del pod permanezca inalterada a lo largo de su ciclo de vida. Kubernetes puede reubicar o reinstanciar los pods en el tiempo de ejecución, lo cual hace que cambie la dirección IP del pod.

Para compensar esta incertidumbre, los servicios garantizan que el tráfico se dirija siempre al pod adecuado dentro del clúster, independientemente del nodo en el que se haya programado. Cada servicio expone una dirección IP y también podría exponer un endpoint DNS, dos elementos que no cambiarán nunca. Los consumidores internos o externos que tengan que comunicarse con un conjunto de pods utilizarán la dirección IP del servicio o su endpoint DNS, un elemento más conocido. De este modo, el servicio funciona como aglutinante para conectar unos pods con otros.

Una implementación utiliza las etiquetas y los selectores para determinar qué pods participarán en una operación de escalado. Cualquier pod cuya etiqueta coincida con el selector definido en el servicio se expondrá a su endpoint. A continuación, un servicio proporciona un equilibrio de cargas básico mediante el direccionamiento del tráfico entre los pods coincidentes.

Un selector es una especie de criterio utilizado para saber qué objetos de Kubernetes tienen un determinado valor de etiqueta. Esta técnica es muy eficaz y garantiza que la vinculación de los objetos no sea rígida. Es posible generar objetos nuevos cuyas etiquetas coincidan con el valor de los selectores. Las etiquetas y los selectores constituyen el mecanismo de agrupación principal de Kubernetes para identificar los componentes pertinentes de una operación.

En el tiempo de ejecución, los pods se pueden ampliar mediante ReplicaSets, para garantizar así que cada implementación ejecute siempre el número deseado de pods. Cada ReplicaSet mantiene en todo momento un conjunto de pods predefinido. Cuando una implementación inicia una operación de escalado, los nuevos pods creados por dicha operación empezarán a recibir tráfico de inmediato.

Kubernetes proporciona tres esquemas para exponer los servicios:

1. **ClusterIP:** concebido para que los pods se comuniquen entre sí dentro del clúster. Por ejemplo, un pod de base de datos expuesto mediante un servicio basado en ClusterIP estará disponible para los pods de servidor web.
2. **NodePort:** se utiliza para exponer un servicio en el mismo puerto en todos los nodos de un clúster. Con un mecanismo de direccionamiento interno, se garantiza que la solicitud se reenvíe al nodo correspondiente de cada pod. Se suele utilizar para los servicios con consumidores externos.
3. **LoadBalancer:** el tipo LoadBalancer amplía el servicio NodePort añadiendo equilibradores de carga de capa 4 (L4) y de capa 7 (L7). Este esquema se suele utilizar con los clústeres que se ejecutan en entornos de nube pública compatibles con el aprovisionamiento automatizado de equilibradores de carga definidos por software.

Cuando varios servicios tienen que compartir un mismo equilibrador de carga o endpoint externo, se recomienda utilizar un controlador de entrada. Este gestiona el acceso externo a los servicios de un clúster —por lo general, HTTP— proporcionando equilibrio de cargas, terminación de capa de sockets seguros (SSL, por sus siglas en inglés) y hosting virtual basado en nombres.

Cada vez se utiliza más el controlador de entrada a la hora de ejecutar cargas de trabajo de

producción en Kubernetes. De este modo, varios microservicios de una misma aplicación pueden utilizar un mismo endpoint, que se expone con un equilibrador de carga, una API Gateway o un controlador de entrega de aplicaciones (ADC, por sus siglas en inglés).

Red y almacenamiento

La computación, las redes y el almacenamiento son las bases de cualquier servicio de infraestructura. En Kubernetes, los nodos constituyen el componente de computación, que proporciona estos recursos básicos a los pods que se ejecutan en los clústeres. Los servicios de red y almacenamiento se proporcionan mediante complementos definidos por software y nativos para contenedores diseñados para Kubernetes.

El componente de red habilita la comunicación entre pods, entre nodo y pod, entre pod y servicio, y entre clientes externos y servicio. Kubernetes sigue un modelo de complementos para implementar la conectividad. Kubenet es el complemento de red predeterminado y resulta muy fácil de configurar. Se suele utilizar junto con un proveedor de servicios en la nube que establece reglas de enrutamiento para la comunicación entre nodos o en entornos de un solo nodo.

Kubernetes admite una gran cantidad de complementos basados en la especificación de la interfaz de red de contenedores (CNI, por sus siglas en inglés), que define la conectividad de red de los contenedores y se ocupa de los recursos de red cuando se elimina el contenedor. Existen numerosas implementaciones de CNI, como Calico, Cil3ium, Contiv y Weave Net, entre otras. La especificación CNI también admite las redes virtuales disponibles en las nubes públicas, lo que permite ampliar la topología de la red y las subredes a los clústeres de Kubernetes.

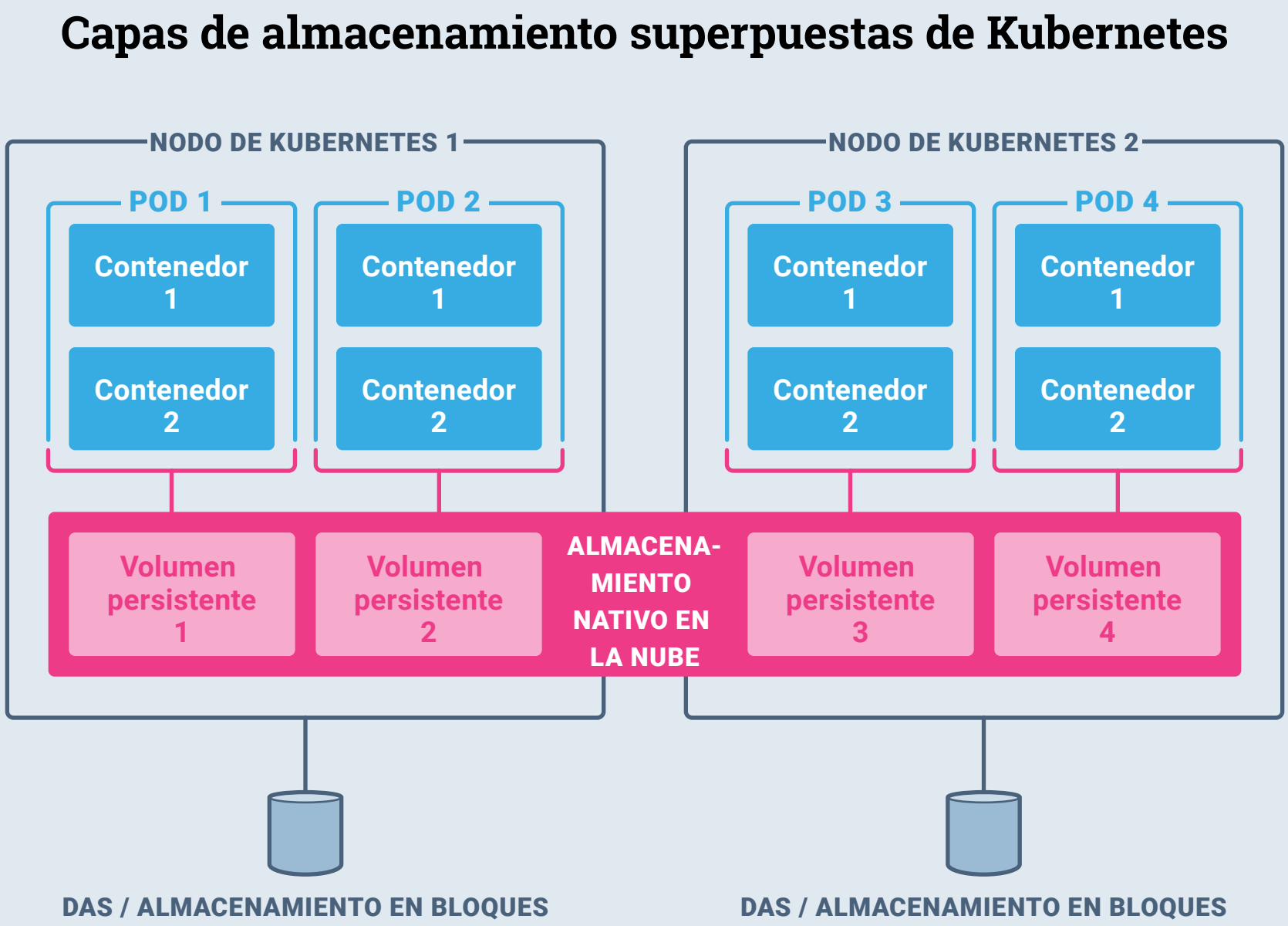
Algunos de los complementos de red compatibles con CNI, como Calico, aplican políticas de enrutamiento estrictas mediante el aislamiento de pods. Aplican a los pods reglas similares a las de un cortafuegos y espacios de nombres de un clúster de Kubernetes.

El almacenamiento persistente se expone a Kubernetes mediante volúmenes persistentes. Los pods consumen los volúmenes mediante reclamaciones de volumen persistentes. Los administradores del almacenamiento proporcionan almacenamiento creando los volúmenes persistentes a partir del almacenamiento conectado a la red (NAS, por sus siglas en inglés)

existente, la red de área de almacenamiento (SAN), el almacenamiento conectado directamente (DAS), las unidades de estado sólido (SSD), la memoria exprés no volátil (NVMe) o las matrices de discos flash. Los desarrolladores y los equipos de DevOps reciben una buena parte de los volúmenes persistentes mediante las reclamaciones de volúmenes persistentes asociadas con los pods.

Kubernetes cuenta con primitivas de almacenamiento para exponer el almacenamiento de los nodos existentes. Una de ellas es un tipo de volumen que hace que el almacenamiento subyacente sea accesible para los pods. Entre los tipos de volumen, se encuentran `emptyDir` y `hostPath`, que se utilizan para ciertos casos de uso: `emptyDir` es para espacio de desecho, mientras que `hostPath` pone los volúmenes locales a disposición de los pods. Pero ambos carecen de alta disponibilidad y tolerancia a los fallos, debido a la vinculación estricta con el nodo. Las capas de almacenamiento superpuestas (véase la figura 1.8) extraen volúmenes de almacenamiento de los dispositivos de bloques, NAS y SAN para exponer el almacenamiento externo a los objetos de Kubernetes.

FIG. 1.8: *Exposición del almacenamiento a los pods y contenedores.*



Fuente: Janakiram MSV

Para ofrecer una alta disponibilidad y capacidades de almacenamiento nativas para contenedores, Kubernetes ofrece una serie de complementos para que los proveedores de almacenamiento puedan exponer sus plataformas a cargas de trabajo basadas en contenedores. El almacenamiento en bloques de los proveedores de nubes públicas, los sistemas de archivos distribuidos basados en NFS y GlusterFS y algunas plataformas de almacenamiento comerciales disponen de complementos incluidos en la distribución ascendente de código abierto de Kubernetes. Los administradores del almacenamiento crean clases de almacenamiento para cada tipo de motor de almacenamiento según su rendimiento y velocidad. Desde estas clases de almacenamiento se pueden crear reclamaciones y volúmenes persistentes para distintos tipos de cargas de trabajo. Por ejemplo, se podría asociar un sistema de gestión de bases de datos relacionales (RDBMS, por sus siglas en inglés) a una clase de almacenamiento con más operaciones de entrada/salida por segundo (IOPS), mientras que un sistema de gestión de contenidos (CMS) podría asociarse a un motor de almacenamiento distribuido mediante una clase de almacenamiento diferente.

De forma similar a lo que ocurre con la CNI, la comunidad de Kubernetes ha definido una serie de especificaciones para el almacenamiento mediante la Container Storage Interface (CSI, por sus siglas en inglés), que fomenta la adopción de un enfoque estándar y portátil para la implementación y el consumo de servicios de almacenamiento por parte de cargas de trabajo basadas en contenedores.

Cómo cumple Kubernetes la promesa de la computación a escala web

Kubernetes, que tiene su linaje en Borg, se ha diseñado para cargas de trabajo a gran escala. Su arquitectura moderna garantiza un aprovechamiento óptimo de los recursos de la infraestructura. Resulta sencillo añadir nodos de trabajo a un clúster existente, prácticamente sin cambios en la configuración. Las cargas de trabajo podrán aprovechar de inmediato los recursos de CPU, memoria y almacenamiento de los nuevos nodos.

La idea de agrupar un conjunto de contenedores relacionados en forma de pod y tratarlos como una unidad de implementación y escala se traduce en un mejor rendimiento. Por ejemplo, al colocar un servidor web y los contenedores de caché en el mismo pod, se reduce la latencia

y se mejora el rendimiento. Los contenedores del pod comparten un mismo contexto de ejecución, lo que les permite utilizar la comunicación entre procesos, con el ahorro de trabajo que esto implica.

Los pods pertenecientes al mismo ReplicaSet y la misma implementación se pueden ampliar con rapidez. Bastan unos segundos para ampliar una implementación a cientos de pods. Los pods se programan en los nodos según los recursos disponibles y el estado deseado de la configuración. Al configurar un Horizontal Pod Autoscaler (HPA), Kubernetes puede reducir y ampliar horizontalmente una implementación de manera automática.

Cuando se ejecuta en entornos de infraestructura flexibles, Kubernetes puede utilizar Cluster Autoscaler para añadir y quitar nodos del clúster. Esta técnica, combinada con HPA, permite gestionar de modo eficiente el escalado automático dinámico tanto de la carga de trabajo como de la infraestructura.

Las soluciones de conectividad ligeras y la detección de servicios de Kubernetes se han diseñado para garantizar la escalabilidad. Son capaces de gestionar decenas de miles de endpoints expuestos por los servicios para su consumo interno y externo.

La comunidad y el ecosistema de Kubernetes se siguen innovando con el objetivo de que la plataforma sea adecuada para cargas de trabajo a gran escala.

Adopción de Kubernetes

Kubernetes se implementa en entornos de producción como motor de orquestación de contenedores, como PaaS y como infraestructura clave para gestionar aplicaciones nativas en la nube. Estos casos de uso no se excluyen entre sí. Los operadores pueden delegar toda la gestión del ciclo de vida de las aplicaciones (ALM, por sus siglas en inglés) a una capa de PaaS basada en Kubernetes. También pueden utilizar una implementación de Kubernetes independiente para gestionar las aplicaciones implementadas con la cadena de herramientas existentes de integración/desarrollo continuos (CI/CD). Los clientes que creen aplicaciones emergentes pueden aprovechar Kubernetes para gestionar el nuevo tipo de aplicaciones nativas en la nube basadas en microservicios mediante soluciones avanzadas, como la aplicación de actualizaciones e

implementaciones de valores controlados.

Los clientes que estén estudiando la posibilidad de adoptar Kubernetes o que lo utilicen para implementaciones de producción pueden elegir entre distintas distribuciones y versiones.

A continuación, facilitamos una lista de las principales distribuciones de Kubernetes disponibles en el mercado.

Distribución ascendente

La distribución ascendente de código abierto de Kubernetes disponible en GitHub se puede implementar en centros de datos, nubes públicas y nubes privadas. La base de código contiene los componentes clave y los elementos esenciales que se necesitan para ejecutar cargas de trabajo en Kubernetes.

Kubernetes incluye una herramienta de implementación llamada kubeadm, que garantiza una experiencia de instalación sencilla a la hora de configurar clústeres basados en CentOS, Red Hat Enterprise Linux, SUSE, Ubuntu y otras distribuciones de Linux.

Los clientes también tienen a su disposición herramientas de implementación automatizada como kops, kubespray y Rancher Kubernetes Engine para la instalación y configuración de clústeres. Estas herramientas ofrecen distintos niveles de de personalización, desde básica hasta avanzada, para configurar los tiempos de ejecución de los contenedores y los complementos de red y almacenamiento.

La distribución ascendente permite instalar Kubernetes de forma gratuita, transparente y totalmente configurable. Sin embargo, los profesionales y las organizaciones que se decanten por esta opción deben tener un buen nivel de conocimientos y competencias.

Distribuciones comerciales

Hay proveedores que ofrecen una versión de Kubernetes personalizada y optimizada, en un paquete que incluye servicios profesionales y de asistencia. Han adoptado un modelo de eficacia probada para comercializar software de código abierto mediante servicios.

Además de acelerar la configuración e implementación de los clústeres de Kubernetes, estas

distribuciones comerciales también prometen parches, revisiones y actualizaciones sencillas a versiones más recientes de la plataforma.

Canonical, D2iQ, HPE, Mirantis, Rancher y VMware son algunos de los proveedores que ofrecen distribuciones comerciales de Kubernetes.

Contenedores como servicio

Kubernetes también está disponible en forma de plataforma gestionada completamente alojada. Ya se trate de un centro de datos empresarial, una infraestructura coubicada o un entorno de nube pública, los proveedores que ofrecen soluciones de contenedores como servicio (CaaS) prometen un entorno de Kubernetes integral respaldado por un acuerdo de nivel de servicio (SLA, por sus siglas en inglés) comercial.

En la actualidad, prácticamente todos los grandes proveedores de servicios en la nube ofrecen alguna solución CaaS. Alibaba Container Service for Kubernetes, Amazon EKS, Azure AKS, DigitalOcean Kubernetes, Google Kubernetes Engine, Huawei Cloud Container Engine e IBM Kubernetes Service son algunos de los ejemplos de CaaS en la nube pública.

Mirantis, NetApp y Platform 9 ofrecen soluciones CaaS para centros de datos y nubes privadas.

Plataforma como servicio

Los clientes que adoptan una PaaS lo hacen principalmente para estandarizar sus entornos de desarrollo e implementación. Al optar por una PaaS basada en Kubernetes, podrán desarrollar tanto aplicaciones de línea de negocio tradicionales como aplicaciones emergentes. Muchos proveedores de PaaS tradicionales han adoptado Kubernetes para ofrecer a los clientes empresariales una plataforma integral.

Las soluciones de PaaS basadas en Kubernetes, que se cimientan en las funciones principales de orquestación de contenedores, permiten gestionar todo el ciclo de vida de las aplicaciones basadas en contenedores.

Red Hat OpenShift y VMware Tanzu Kubernetes Grid son ejemplos de soluciones PaaS basadas en Kubernetes.

Kubernetes como plano de control universal

Kubernetes se está posicionando como uno de los mejores planos de control en el contexto de la infraestructura y las aplicaciones modernas. Su eficaz programador, diseñado en un principio para ocuparse de colocar los pods en los nodos adecuados, es bastante ampliable. Puede solucionar muchos de los problemas existentes en los sistemas distribuidos tradicionales.

Kubernetes se está convirtiendo rápidamente en el plano de control preferido para programar y gestionar tareas en entornos muy distribuidos. Entre dichas tareas, se encuentran la implementación de máquinas virtuales en hosts físicos, la colocación de contenedores en dispositivos perimetrales e incluso la ampliación del plano de control a otros programadores como entornos sin servidor.

Servidores físicos, máquinas virtuales, dispositivos de Internet de las cosas (IdC), servicios en la nube gestionados... Kubernetes no se limita a los contenedores y pods, sino que ayuda a afrontar diversos retos relacionados con el aprovisionamiento y la programación.

Veamos a continuación varios ejemplos.

Crossplane

El objetivo de Crossplane es estandarizar la gestión de las aplicaciones y la infraestructura con el mismo sistema de automatización y configuración declarativo y centrado en las API del que Kubernetes fue pionero. Se trata de un plano de control unificado que se integra a la perfección con las herramientas y los sistemas existentes y que facilita la configuración de políticas, cuotas e informes de seguimiento.

Crossplane funciona como puente entre Kubernetes y las cargas de trabajo tradicionales, como las bases de datos, e incluso los servicios gestionados en la nube pública. DevOps puede declarar recursos externos utilizando la misma especificación YAML, junto con las aplicaciones nativas de Kubernetes. Este método fomenta la configuración como código, pues amplía el control de versiones, la integración continua y la implementación a los recursos que no son de Kubernetes.

K3s

K3s, de Rancher, es una distribución de Kubernetes certificada diseñada para las cargas de trabajo de producción que se ejecutan en entornos con muchas restricciones, como las implementaciones de computación perimetral y de IdC.

K3s se puede implementar en la mayoría de las máquinas virtuales de la nube pública, o incluso en un dispositivo Raspberry Pi. Además de garantizar la plena compatibilidad y el cumplimiento normativo de las pruebas de conformidad con Kubernetes de la CNCF, su arquitectura también está optimizada para las implementaciones remotas desatendidas en dispositivos con recursos limitados.

K3s hace que Kubernetes sea accesible y ligero, lo que permite situarlo en la capa de la computación perimetral.

KubeEdge

Durante el evento KubeCon+CloudNativeCon celebrado en Seattle en 2018, Huawei presentó KubeEdge, el proyecto oficial para llevar las posibilidades de Kubernetes al perímetro.

KubeEdge se basa en el Intelligent Edge Fabric (IEF) de Huawei, una plataforma perimetral de IdC comercial basada en la PaaS de IdC de Huawei. Gran parte del IEF se ha modificado y convertido en código abierto para KubeEdge. En su versión 1.3, KubeEdge ofrece una buena estabilidad y aborda los principales casos de uso relacionados con el IdC y el perímetro. Se puede instalar en una distribución de Linux compatible y en un dispositivo ARM como Raspberry Pi.

El proyecto KubeEdge forma parte del sandbox de la CNCF.

KubeVirt

KubeVirt, un complemento de gestión de máquinas virtuales para Kubernetes, se ha concebido para permitir a los usuarios ejecutar MV además de contenedores en sus clústeres de Kubernetes u OpenShift. Amplía Kubernetes añadiendo tipos de recursos para MV y conjuntos de MV mediante la API CustomResourceDefinitions (CRD) de Kubernetes. Las MV de KubeVirt se ejecutan en pods de Kubernetes normales, donde tienen acceso al almacenamiento y las redes de pod

estándar, y se pueden gestionar con herramientas estándar de Kubernetes como `kubectl`.

KubeVirt también forma parte del sandbox de la CNCF.

Virtual Kubelet

El proyecto Virtual Kubelet de Microsoft es la ampliación más interesante del agente Kubelet y la API de Kubernetes. Virtual Kubelet es un agente que se ejecuta en un entorno externo registrado como nodo dentro del clúster de Kubernetes. El agente crea un recurso de nodo mediante la API de Kubernetes. Recurre a los conceptos de contaminaciones y tolerancias para programar pods en un entorno externo llamando a su API nativa.

Virtual Kubelet funciona con Azure Container Instances, Azure IoT Edge y el plano de control AWS Fargate.

Aunque Kubernetes empezó sin grandes pretensiones en la orquestación de contenedores, pronto evolucionó para convertirse en el sistema operativo de la nube y el perímetro. En la actualidad, es la base de la infraestructura moderna en centros de datos, nubes híbridas, nubes públicas y entornos de varias nubes.

En el próximo capítulo, analizaremos el ecosistema de Kubernetes, que no deja de crecer, y el sector de las soluciones nativas en la nube.

Las arquitecturas de autoservicio y el operador de Kubernetes para Cassandra

A lo largo de la última década, hemos visto un auge de las bases de datos distribuidas —como la plataforma de código abierto Apache Cassandra— concebidas para complementar las funciones de Kubernetes y su arquitectura con escalabilidad horizontal. Esta tendencia ha dado paso a la era de las arquitecturas cien por cien de autoservicio, de las que hablamos en este podcast con Kathryn Erickson y Patrick McFadin de DataStax, una empresa especializada en plataformas de gestión de bases de datos NoSQL nativas en la nube.

DataStax ofrece una guía para Cassandra a través de un nuevo operador de Kubernetes, que abstrae las capas de la base de datos para que los desarrolladores puedan centrarse en lo que mejor se les da: programar.

Gracias a las arquitecturas de autoservicio, los desarrolladores tienen más tiempo para buscar nuevas formas de utilizar los datos. O, en palabras del propio McFadin: «Ahora, la carga cognitiva que representan las bases de datos para los desarrolladores es menor que antes».

Escuchar en SoundCloud **Ver en YouTube**

Kathryn Erickson dirige la estrategia de negocio de las soluciones de autoservicio para DataStax.

Patrick McFadin es evangelista jefe para Apache Cassandra y vicepresidente de relaciones con los desarrolladores de DataStax.

La observabilidad basada en la IA disminuye la complejidad de Kubernetes

Kubernetes ha hecho realidad las aplicaciones con escalabilidad horizontal en varios entornos en la nube. Pero también ha introducido una enorme complejidad en los departamentos de TI.

Hace poco, Andreas Grabner, activista de la metodología DevOps de la plataforma de inteligencia de software Dynatrace, apuntaba que, en los entornos empresariales de Kubernetes, «hay que tener en cuenta miles de millones de interdependencias». Ha leído bien: miles de millones.

En este podcast, Grabner explica cómo la tecnología de inteligencia artificial simplifica toda esta complejidad. También hablamos de varios casos de uso de la observabilidad con IA para los operadores y desarrolladores, el valor que aportan los datos de telemetría a los equipos de operaciones encargados de gestionar Kubernetes y los cambios culturales que se han producido en los equipos de desarrollo a lo largo de la era Kubernetes.

Escuchar en SoundCloud **Ver en YouTube**

Andreas Grabner es activista de la metodología DevOps en Dynatrace. Ayuda a los desarrolladores, responsables de pruebas y profesionales de operaciones y xOps a ser más eficientes en su trabajo con ayuda de Dynatrace.

Cómo adaptar las aplicaciones orientadas a los datos a las arquitecturas de Kubernetes

Al igual que los microservicios, Kubernetes está «sin dependencia fija» por naturaleza. Los componentes son escasos, tienen un único propósito y funcionan por separado. En cambio, actualizar un código integrado e interdependiente cuando se introducen cambios en la aplicación puede resultar más complejo.

Conversamos con Nanda Vijaydev, destacada tecnóloga y científica de datos principal de HPE, sobre cómo gestionar aplicaciones orientadas a los datos en Kubernetes en este contexto de ausencia de dependencia fija requiere, en muchos sentidos, un mayor cuidado y atención.

«¿Cómo implementar a gran escala y de forma dinámica algo que está concebido como un modelo de datos?», plantea. «La tecnología está evolucionando en esa dirección precisamente».

Gracias a la adquisición de MapR Technologies, BlueData y Cloud Technology Partners (CTP), entre otras, HPE ha reforzado su capacidad de ayudar a las organizaciones a responder a los desafíos asociados con la necesidad de integrar las cargas de trabajo orientadas a los datos que se ejecutan en Kubernetes. Su plataforma de software permite a los usuarios percibir la gestión de datos, más que como un problema de Kubernetes, como un problema de almacenamiento o de red.

Escuchar en SoundCloud

Nanda Vijaydev es una tecnóloga destacada y científica de datos principal de HPE, donde utiliza tecnologías como Kubernetes, TensorFlow, H2O y Spark para crear soluciones que respondan a las necesidades de aprendizaje automático y aprendizaje profundo de las empresas.

CAPÍTULO 2

Diseño del ecosistema de Kubernetes

Kubernetes se ha alzado vencedor en la batalla de las herramientas de orquestación por convertirse en la plataforma de gestión de contenedores preferida. Se ha erigido como un software de infraestructura unificado y homogéneo que se ejecuta en centros de datos empresariales, plataformas en la nube pública, plataformas en la nube híbrida, dispositivos de infraestructura hiperconvergente e incluso en el perímetro.

Kubernetes constituye la base de la infraestructura moderna que ejecuta cargas de trabajo contemporáneas, a las que se suele llamar «aplicaciones nativas en la nube». La capa de la infraestructura principal se complementa y refuerza con software comercial y proyectos de código abierto que conforman el conjunto de tecnologías nativas en la nube.

En este capítulo, ofrecemos una visión de conjunto de las tecnologías nativas en la nube, para luego analizar a fondo los componentes básicos de la tecnología nativa en la nube.

Nos detendremos en cada uno de los elementos del entorno, desde el sistema operativo hasta la experiencia de los desarrolladores, para examinar más de cerca los principales proyectos de código abierto y soluciones comerciales que ponen la computación nativa en la nube al alcance de las empresas y los usuarios.

En este capítulo nos centramos exclusivamente en el núcleo de Kubernetes. Por ejemplo, los sistemas de bases de datos forman parte del ecosistema más amplio de Kubernetes. En concreto, las bases de datos como servicio (DBaaS, por sus siglas en inglés) han ganado importancia en los últimos años, y ahora abarcan productos como Amazon Aurora, Azure SQL Database, MongoDB Atlas y Redis Cloud Essentials. Sin embargo, la base de datos no es un componente central de la tecnología nativa en la nube, sino que se trata de una carga de trabajo. El componente clave detrás de las bases de datos y las cargas de trabajo con estado es el almacenamiento, una parte central de la tecnología nativa en la nube de la que nos ocupamos a continuación.

El auge de las soluciones nativas en la nube y los contenedores como servicio (CaaS)

Según The Linux Foundation, la computación nativa en la nube utiliza una serie de programas de software de código abierto para implementar las aplicaciones como microservicios, empaquetando cada parte en un contenedor distinto y orquestando de forma dinámica dichos contenedores para optimizar el uso de los recursos.

La expresión «nativo en la nube» se utiliza para describir los entornos basados en contenedores. Las tecnologías nativas en la nube sirven para desarrollar aplicaciones creadas con servicios empaquetados en contenedores, implementadas como microservicios y gestionadas en una infraestructura flexible mediante procesos de DevOps ágiles y flujos de trabajo de entrega continua.

Uno de los atributos clave de la computación nativa en la nube es la portabilidad, que solo es posible si la infraestructura es homogénea en los distintos entornos. Por eso, Kubernetes se convierte en el denominador común mínimo de la infraestructura y en la base de las soluciones nativas en la nube.

Si bien Kubernetes es un elemento importante de la tecnología nativa en la nube, los desarrolladores y los ingenieros de DevOps necesitan software adicional para implementar, ampliar y gestionar las aplicaciones modernas. Hay proveedores de plataformas, como Red Hat y VMware, que ofrecen plataformas integrales basadas en Kubernetes. Los proveedores de nube pública —como Amazon Web Services (AWS), Google Cloud Platform (GCP) y Microsoft Azure— ofrecen servicios gestionados basados en Kubernetes que se ejecutan en la infraestructura de computación, almacenamiento y redes ya existente.

Las plataformas de gestión de contenedores integradas basadas en Kubernetes han creado una nueva categoría de modelo de entrega de aplicaciones: los contenedores como servicio (CaaS). De forma similar a lo que ocurre con una plataforma como servicio (PaaS), la plataforma de gestión de contenedores se puede implementar detrás del cortafuegos que se ejecuta en un centro de datos empresarial o bien consumir como oferta de servicios en la nube gestionados.

Si las organizaciones utilizan los CaaS como tejido común del centro de datos y la nube pública, podrán crear aplicaciones híbridas para conectar de forma segura los recursos internos a la nube pública. En poco tiempo, los CaaS están contribuyendo a la adopción de los entornos de nube híbrida y las implementaciones de varias nubes, ya que los desarrolladores y operadores pueden transferir aplicaciones de un entorno a otro con facilidad.

Atributos clave de una plataforma de gestión de contenedores

Independientemente de dónde se implemente, la plataforma de gestión de contenedores debe cumplir los siguientes requisitos:

- **Coherencia:** los desarrolladores y operadores esperan una experiencia homogénea en la nube pública y en los entornos locales.
- **Procesos de DevOps:** la plataforma debe integrar prácticas de DevOps de eficacia probada que garanticen la entrega rápida de software.
- **Seguridad:** la plataforma de gestión de contenedores debe garantizar la seguridad de la infraestructura y las aplicaciones. Para ello, tiene que ser capaz de detectar las vulnerabilidades antes de implementar las aplicaciones y supervisar constantemente la infraestructura para encontrar posibles infracciones.
- **Fiabilidad de la infraestructura:** se debe adoptar un modelo de prestación de servicios basado en acuerdos de nivel de servicio (SLA) que garantice el máximo tiempo de actividad de la infraestructura y la plataforma.
- **Alta disponibilidad de las cargas de trabajo:** además de la infraestructura, también las aplicaciones empresariales implementadas en la plataforma deben garantizar una alta disponibilidad.
- **Observabilidad:** la plataforma debe garantizar la visibilidad de la infraestructura, los recursos y las aplicaciones. Para ello, recopilará métricas, eventos, logs y trazas de todas las soluciones y los almacenará en una ubicación centralizada.
- **Gestión de varios inquilinos y basada en políticas:** de forma opcional, la plataforma de

gestión de contenedores puede proporcionar un aislamiento estricto entre los distintos inquilinos que utilizan la plataforma. La implementación y gestión de las aplicaciones deberían estar sujetas a políticas bien definidas.

Plataformas de gestión de contenedores: visión de conjunto

El conjunto de tecnologías nativas en la nube modernas que se distribuye en forma de plataforma integrada de gestión de contenedores está formado por varios componentes básicos. Algunos de ellos están disponibles como proyectos de código abierto, mientras que otros son productos comerciales vendidos por proveedores de software independientes.

La capa inferior constituye la infraestructura física del clúster, formada por los componentes de computación, almacenamiento y redes. La plataforma añade varias capas de abstracción para aprovechar al máximo la infraestructura física subyacente.

FIG. 2.1: *Tecnología nativa en la nube.*

Fuente: Janakiram MSV

Veamos de forma más detallada cada una de estas capas.

Sistema operativo optimizado para contenedores

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Fedora CoreOS	Red Hat	Flatcar Container Linux	No presentado
Talos	Talos Systems	RancherOS	No presentado

Los contenedores han redefinido la función de los sistemas operativos (SO). Como gran parte del trabajo pesado se ha trasladado a los tiempos de ejecución de los contenedores, el sistema operativo se ha convertido en una fina capa que proporciona acceso a los recursos físicos. Este cambio ha dado lugar a la aparición de un nuevo tipo de sistemas operativos, los sistemas operativos optimizados para contenedores (COS, por sus siglas en inglés).

En comparación con un SO tradicional, un COS es más ligero y ocupa una superficie mucho más reducida. Contiene los componentes imprescindibles para ejecutar el tiempo de ejecución del contenedor. Para mantener una implementación de CaaS, es fundamental elegir el COS adecuado.

Para implementar el COS, los clientes pueden elegir entre Fedora CoreOS (de Red Hat), Talos (de Talos Systems), Flatcar Container Linux (de Kinvolk GmbH) y RancherOS (de Rancher Labs, que en julio de 2020 estaba siendo adquirido por SUSE).

La mayoría de los proveedores ofrecen la posibilidad de adquirir un plan de suscripción comercial con actualizaciones periódicas, revisiones y asistencia profesional.

Tiempo de ejecución del contenedor

El tiempo de ejecución del contenedor se encarga de gestionar el ciclo de vida de un contenedor, para lo cual proporciona el entorno de ejecución y funciona como interfaz entre la carga de trabajo y el sistema operativo host.

En 2015, The Linux Foundation lanzó la Open Container Initiative (OCI) para garantizar la paridad entre las implementaciones del tiempo de ejecución del contenedor. En la actualidad, la OCI define

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Docker Engine Enterprise	Mirantis	containerd	Graduado
--	--	CRI-O	En incubación
--	--	Docker-CE	No presentado
--	--	Kata Containers	No presentado
--	--	runC	No presentado

dos especificaciones: la Runtime Specification (runtime-spec) y la Image Format Specification (image-spec).

Según se explica en el sitio web de la OCI, la primera especifica cómo ejecutar un «paquete de sistema de archivos» que se desempaqueta en un disco. En términos generales, una implementación OCI descarga una imagen OCI y, a continuación, la desempaqueta en un paquete de sistemas de archivos del tiempo de ejecución OCI.

La Image Format Specification especifica cómo crear una imagen OCI —algo que, en general, se hará mediante un sistema de compilación— y cómo producir un manifiesto de imagen, un sistema de archivos serializado (capa) y una configuración de imagen.

Desde que Mirantis adquirió Docker Enterprise, es Mirantis quien vende la edición comercial de Docker Engine (Docker Engine Enterprise), acompañada de asistencia de clase empresarial y servicios profesionales.

El proyecto containerd ha evolucionado hasta convertirse en un estándar del sector para el tiempo de ejecución del contenedor. Es un proyecto avalado por la CNCF, con el estado de «graduado», que se utiliza en numerosos entornos de producción. CRI-O es un proyecto incubado en la CNCF en el que la comunidad participa activamente.

Docker Engine (hoy, Docker-CE) es uno de los tiempos de ejecución del contenedor más utilizados por las plataformas de gestión de contenedores. Frakti (un método más antiguo) es un tiempo de ejecución del contenedor para Kubernetes basado en hipervisor que brinda un aislamiento más estricto mediante la ejecución de pods en MV específicas. Aparte de las opciones mencionadas,

hay otras como Kata Containers y runC.

Almacenamiento nativo para contenedores

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Red Hat OpenShift Container Storage	Red Hat	Ceph	No presentado
Kubera	MayaData	Longhorn	Sandbox
Portworx	Portworx	OpenEBS	Sandbox
Robin	Robin Systems	Rook	En incubación
StorageOS	StorageOS	--	--
Trident	NetApp	--	--

El almacenamiento es uno de los componentes más cruciales de una plataforma CaaS. El almacenamiento nativo para contenedores expone los servicios de almacenamiento subyacentes a los contenedores y microservicios. Al igual que el almacenamiento definido por software, agrega y agrupa recursos de almacenamiento procedentes de medios diversos.

El almacenamiento nativo para contenedores permite la ejecución de cargas de trabajo con estado dentro de los contenedores gracias al aprovisionamiento de volúmenes persistentes. Junto con primitivas de Kubernetes como StatefulSets, garantiza la fiabilidad y estabilidad necesarias para ejecutar cargas de trabajo cruciales en entornos de producción.

Aunque Kubernetes admita sistemas de archivos distribuidos tradicionales como los sistemas de archivos de red (NFS) y GlusterFS, recomendamos utilizar un tejido de almacenamiento que tenga en cuenta los contenedores y se haya diseñado para responder a los requisitos de las cargas de trabajo con estado que se ejecutan en los entornos de producción. Existen diversos proyectos de código abierto e implementaciones comerciales para elegir.

El ecosistema nativo en la nube ha definido una serie de especificaciones para el almacenamiento mediante la Container Storage Interface (CSI), que fomenta la adopción de un enfoque estándar y portátil para la implementación y el consumo de servicios de almacenamiento por parte de cargas de trabajo basadas en contenedores.

Ceph, Longhorn, OpenEBS y Rook son proyectos de código abierto de almacenamiento nativo para contenedores, mientras que Kubera (de MayaData), Trident (de NetApp), Portworx, Red Hat OpenShift Container Storage, Robin (de Robin System) y StorageOS son productos comerciales que incluyen un servicio de asistencia.

También hay proveedores tradicionales como NetApp, Pure Storage y VMware que ofrecen complementos de almacenamiento para Kubernetes.

Respuestas a los desafíos relacionados con la infraestructura

Con las soluciones de Kubernetes gestionadas, se pueden reducir tanto la complejidad de las implementaciones de contenedores de gran tamaño como las competencias necesarias para gestionarlas. Cuando los profesionales de TI evalúan las hojas de ruta de sus iniciativas tecnológicas, uno de los criterios más importantes es la optimización de la infraestructura en que se basan las cargas de trabajo de Kubernetes.

Pasemos a analizar los datos obtenidos en la encuesta de la CNCF de 2019 sobre los planes de adopción de Kubernetes actuales y futuros y sobre las dificultades relacionadas con los contenedores que afrontan los usuarios de Kubernetes. La encuesta revela que la adopción temprana dependía de las relaciones forjadas con los proveedores elegidos en ese momento. Sin embargo, el nivel de satisfacción variaba.

Muchos usuarios de Kubernetes daban prioridad a los proveedores de soluciones en la nube y de almacenamiento con los que contaban en ese momento, pero no les resultaba fácil hacer cribado. Al menos el 5 % de los usuarios de Kubernetes evaluaron las 38 opciones contempladas en la encuesta.

Con el auge de los servicios de Kubernetes gestionados, los proveedores de servicios en la nube empezaron a exponer el almacenamiento en bloques mediante las clases de almacenamiento y el aprovisionamiento dinámico. Los clientes podían conectar volúmenes de Amazon Elastic Block Store (EBS) a AWS, discos gestionados de Azure, Google Persistent Disks y con los nodos de trabajo de Kubernetes que se ejecutaban en AWS, GCP y Microsoft Azure. Esto jugaba a favor de los proveedores de servicios en la nube.

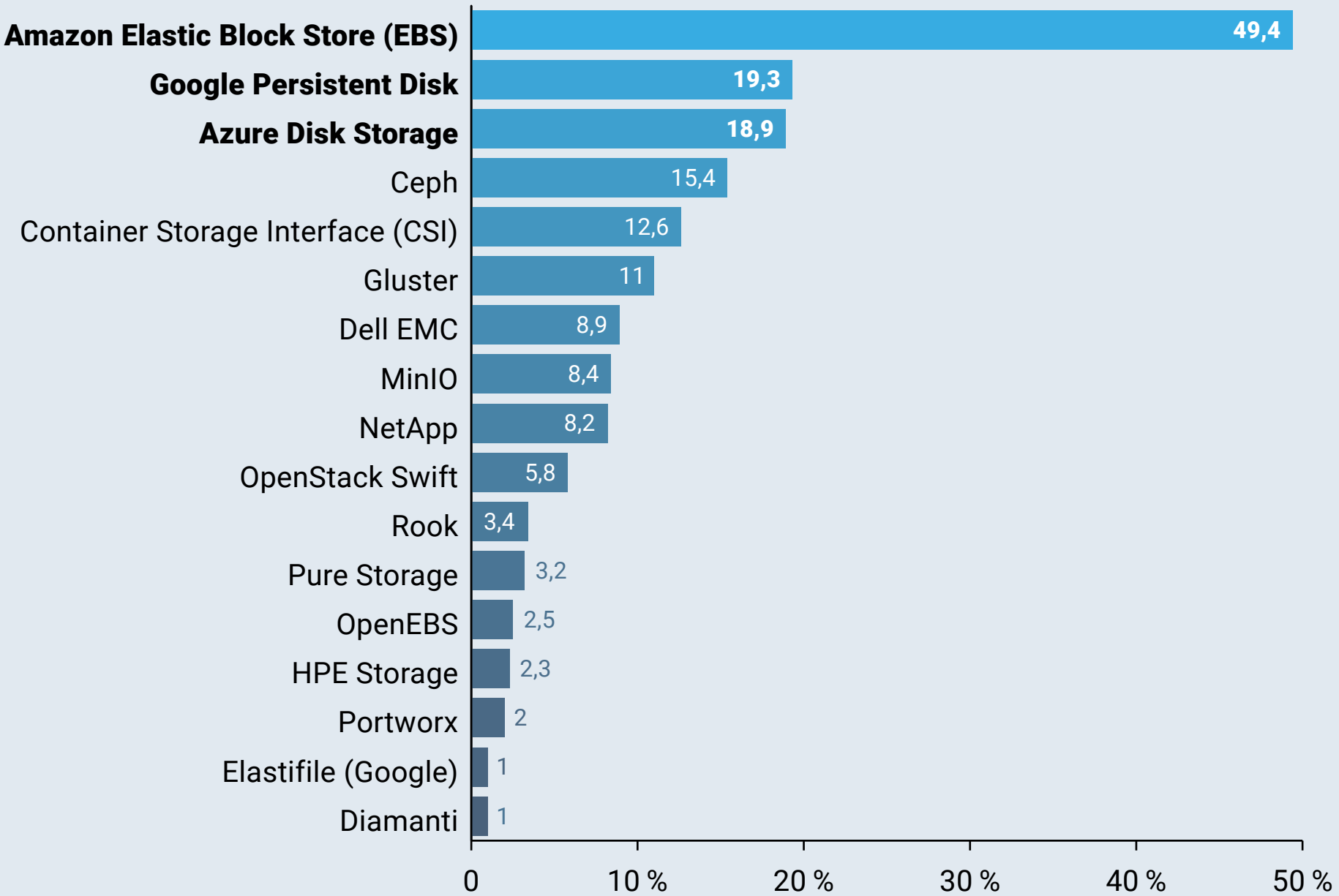


FIG. 2.2: *Los sistemas de archivos como Ceph se suelen considerar competidores del almacenamiento en la nube, a diferencia de los productos que ofrecen las empresas de almacenamiento más tradicionales.*

Cuando se preguntaba a los usuarios de Kubernetes por el almacenamiento nativo en la nube que utilizaban (véase la figura 2.2 a continuación), las soluciones más mencionadas eran Amazon EBS, Google Persistent Disk y Azure Disk Storage. En muchos casos, los StatefulSets permitían a las cargas de trabajo del clúster acceder al almacenamiento en bloques que ofrecía el proveedor de servicios en la nube. Aunque gozaban de una amplia adopción, las soluciones de almacenamiento en bloques que ofrecían los proveedores de servicios en la nube no se habían diseñado específicamente para las cargas de trabajo de Kubernetes.

Las siguientes de la lista eran Ceph, CSI y Gluster; y el 37 % de los usuarios de Gluster utilizaban también Ceph. Ceph y Gluster son sistemas de archivos distribuidos que añaden una capa de persistencia entre distintos nodos. Sin embargo, no están bien integrados en las herramientas y el flujo de trabajo de Kubernetes, por lo que es posible que a los administradores del almacenamiento les resulte más difícil mantenerlos y configurarlos.

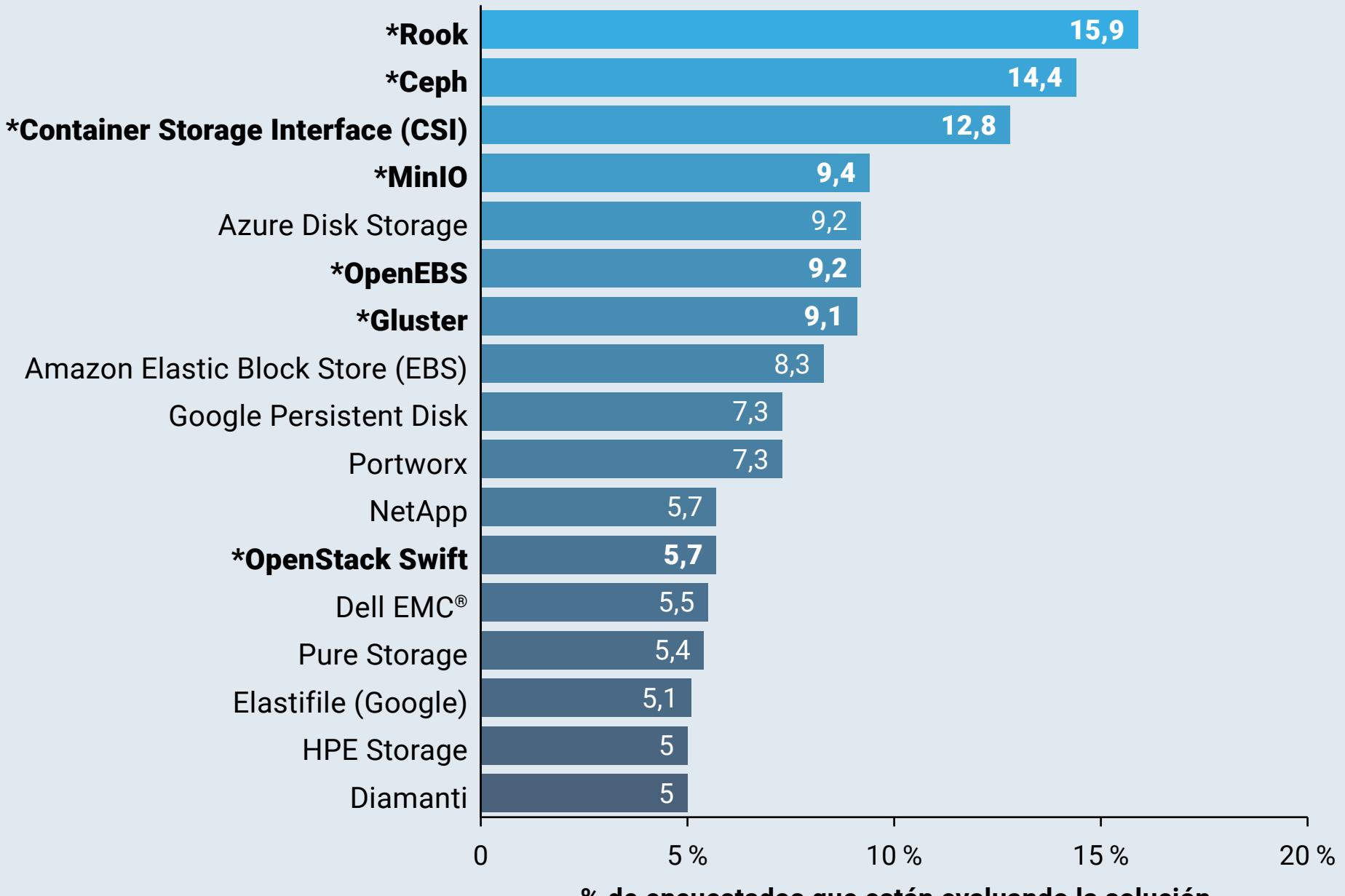
Se posicionaban peor en la lista los productos de empresas consolidadas centradas en el almacenamiento, como Dell EMC, NetApp y Pure Storage. Al principio, Kubernetes había integrado complementos de volumen para conectar con los sistemas de almacenamiento internos de estas empresas; pero, lamentablemente, la distribución ascendente de Kubernetes se saturó, lo que significaba que cualquier actualización o cambio en un complemento, por pequeño que fuese, implicaba volver a crear y compilar todo el código.

Entre los clientes de empresas de almacenamiento tradicionales, la probabilidad de sufrir problemas con el almacenamiento era considerablemente mayor. Por ejemplo, el 46 % de los clientes de Pure Storage tenían dificultades en la gestión del almacenamiento relacionado con los contenedores, mientras que el porcentaje se reducía al 27 % en el caso del usuario medio de Kubernetes. De todas formas, había motivos para la esperanza, pues el 13 % utilizaba Container Storage Interface (CSI). En 2019, CSI pasó a estar disponible de forma generalizada para Kubernetes, lo que elimina la dificultad que supone la necesidad constante de integración ascendente. Los proveedores de almacenamiento tradicionales, los proveedores de servicios en la nube y las empresas centradas en el almacenamiento de contenedores, como Portworx, se están pasando a CSI.

Quienes evaluaban nuevas posibilidades para resolver las dificultades que les planteaba el almacenamiento de contenedores tenían en mente la CSI. Aunque en general solo el 13 % de los usuarios de Kubernetes consideraban la CSI, el porcentaje subía hasta el 22 % entre quienes tenían problemas con el almacenamiento.

Si bien había quien valoraba la posibilidad de recurrir a empresas consolidadas, los proyectos de código abierto eran las opciones prioritarias para quienes buscaban nuevas soluciones de almacenamiento (véase la figura 2.3). En comparación con el encuestado medio, el 26,6 % de los usuarios de Kubernetes que tenían problemas con el almacenamiento eran más proclives a considerar Rook (un 25,9 % frente a un 15,9 %), Ceph (un 23,4 % frente a un 14,4 %), Gluster (un 14,6 % frente a un 9,1 %), OpenEBS (un 14,6 % frente a un 9,2 %) y MinIO (un 12,8 % frente a un 9,4 %). Era evidente que el motivo de estas iniciativas de código abierto no era la necesidad de vender hardware.

Tanto para las empresas de almacenamiento tradicionales como para las nuevas ofertas de



Fuente: Análisis de The New Stack de la encuesta de la CNCF realizada en 2019. P: Indique si su empresa/organización utiliza en el entorno de producción alguno de estos proyectos de almacenamiento nativo en la nube o está evaluando su adopción. ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan.Solo se muestran los datos relativos a las soluciones utilizadas por un mínimo del 1 % de los encuestados que tienen al menos un clúster de Kubernetes (n=1149).

FIG. 2.3: *Al analizar más a fondo las respuestas de los usuarios de Kubernetes que mencionaron dificultades con el almacenamiento (no se muestran los datos), se observa que la probabilidad de considerar la adopción de Rook, Ceph y OpenEBS era un 50 % mayor entre quienes tenían dificultades con el almacenamiento. Los tres tienen controladores de CSI.*

almacenamiento nativo en la nube, era más probable que los usuarios mencionasen dificultades relacionadas con el almacenamiento. Sin embargo, al implementar nuevos métodos como CSI, las empresas de almacenamiento tradicionales respondían a las preocupaciones de sus clientes. Aunque muchos usuarios de soluciones nuevas —como OpenEBS (de MayaData), Minio y Portworx— declararon tener dificultades con el almacenamiento, era probable que se refiriesen a problemas al conectar sus almacenes de datos antiguos.

Los problemas de implementación eran frecuentes entre quienes adoptaban soluciones destacadas de terceros en una fase temprana. Conforme pase el tiempo, será interesante evaluar la eficacia de los nuevos actores, ya que podría afectar a la capacidad de las empresas de almacenamiento tradicionales y en la nube de retener su clientela en este segmento.

Redes nativas para contenedores

De modo similar a lo que ocurre con el almacenamiento nativo para contenedores, la red nativa para contenedores abstrae la infraestructura de red física para exponer una red plana a los contenedores. Está estrechamente integrada con Kubernetes para abordar las dificultades propias de la comunicación externa, de pod a pod, de nodo a nodo y de pod a servicio.

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Calico Enterprise	Tigera	Cilium	No presentado
Weave Net (contactar al proveedor si se desea una suscripción empresarial)	Weaveworks	Contiv	No presentado
--	--	Flannel	No presentado
--	--	Project Calico	No presentado
--	--	Tungsten Fabric	No presentado
--	--	Weave Net	No presentado

Kubernetes admite una gran cantidad de complementos basados en la especificación de la interfaz de red de contenedores (CNI), que define la conectividad de red de los contenedores y se ocupa de los recursos de red cuando se elimina el contenedor. El proyecto de la CNI forma parte de los proyectos incubados en la CNCF.

Las redes nativas para contenedores van más allá de la conectividad básica: aplican de forma dinámica las reglas de seguridad de la red. Mediante una política predefinida, se puede configurar un control detallado de las comunicaciones entre contenedores, pods y nodos.

Para mantener y proteger la plataforma CaaS, resulta crucial elegir las soluciones de conectividad adecuadas. Para ello, los clientes tienen a su disposición proyectos de código abierto como Cilium, Contiv, Flannel, Project Calico, Tungsten Fabric y Weave Net. En cuanto a las soluciones comerciales, Tigera ofrece Calico Enterprise, y es posible adquirir una suscripción empresarial de Weave Net contactando a Weaveworks.

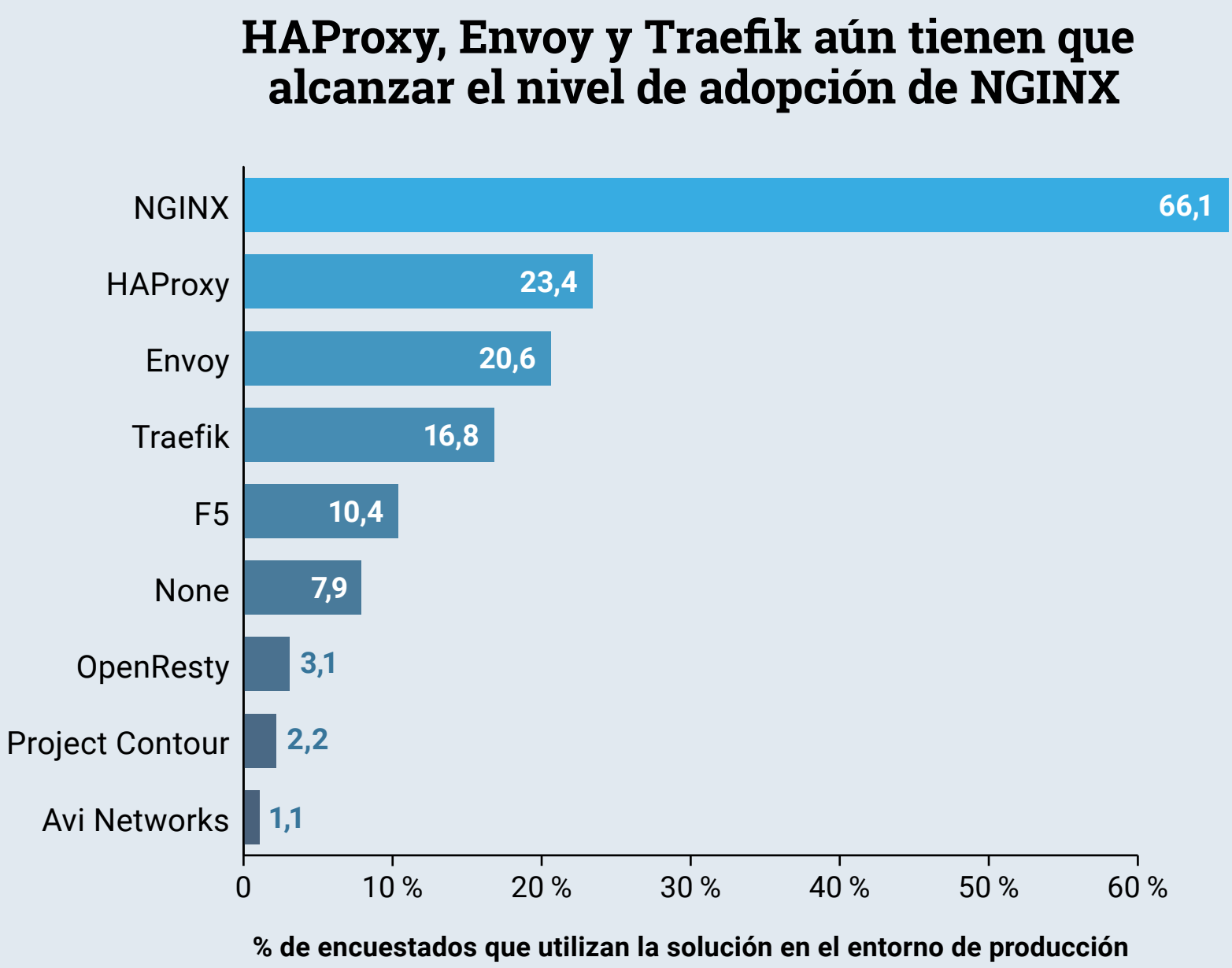
Las soluciones de CaaS gestionadas de proveedores de nube pública ofrecen una integración perfecta con las soluciones de conectividad virtual existentes. Por ejemplo, AWS tiene un

complemento de CNI para Amazon Elastic Kubernetes Service (EKS) basado en Amazon Virtual Private Cloud (VPC), mientras que Microsoft ha creado Azure Virtual Network Container Networking Interface (CNI) para Azure Kubernetes Service (AKS).

Volvamos ahora a los datos de la encuesta de la CNCF de 2019, esta vez en lo que se refiere a la conectividad.

La conectividad es un desafío que ha ido perdiendo importancia con el paso de los años, aunque los usuarios de Kubernetes seguían evaluando varios proveedores de soluciones de entrada. De hecho, los usuarios de Kubernetes tenían, de media, 1,5 proveedores de soluciones de entrada, mientras que el 28 % de los encuestados que habían mencionado dificultades con la conectividad tenían tres proveedores de este tipo por término medio. NGNIX estaba presente en el 66 % de los entornos Kubernetes, pero por sí solo no bastaba para responder a las necesidades de todos los usuarios. La adopción de HAProxy y Envoy era más baja, pero aumentaba entre aquellos que

FIG. 2.4: *A pesar de su relación con la entrada, el uso de NGINX con Kubernetes afecta poco a las dificultades relativas a la conectividad.*



Fuente: Análisis de The New Stack de la encuesta de la CNCF realizada en 2019. QP: ¿A qué proveedor(es) de soluciones de entrada de Kubernetes (por ejemplo, proxy de servicios) recurre? Seleccione todas las respuestas que correspondan. ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan. Solo se muestran los datos relativos a las soluciones utilizadas por un mínimo del 1 % de los encuestados que tienen al menos un clúster de Kubernetes (n=1197).

tenían dificultades con la conectividad.

Podemos concluir que, para solucionar problemas, se adoptaban soluciones que llevaban menos tiempo en el mercado. En el futuro, cabe esperar que los productos se diferencien según los protocolos que admiten y según la presencia o ausencia de una API Gateway.

Seguridad y endurecimiento

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Aqua	Aqua Security Software	Clair	No presentado
Calico Enterprise	Tigera	Falco	En incubación
Prisma Cloud	Palo Alto Networks	Open Policy Agent (OPA)	En incubación
Snyk	Snyk	Snyk (código abierto)	No presentado
StackRox	StackRox	--	--

Para proteger las plataformas de gestión de contenedores, se endurece la red, se escanean las imágenes de contenedor y se realizan periódicamente evaluaciones de riesgos y detecciones de amenazas.

La seguridad de la red es una parte importante del endurecimiento de la plataforma. El departamento de TI debería invertir en una capa de red Zero Trust (confianza cero) para satisfacer los requisitos de seguridad y cumplimiento normativo. Estas redes aplican un control de políticas detallado en varios puntos de la infraestructura, realizan análisis en busca de anomalías y cifran y autorizan el tráfico entre microservicios utilizando para ello protocolos seguros como la seguridad de la capa de transporte mutua (mTLS, por sus siglas en inglés). Calico Enterprise (de Tigera) es uno de los principales proveedores de redes Zero Trust (confianza cero) para Kubernetes y plataformas CaaS.

Una plataforma de contenedores segura implementa las siguientes técnicas para garantizar que la infraestructura y las aplicaciones estén bien protegidas:

- Una plataforma de contenedores segura utiliza **imágenes de confianza** que hayan sido firmadas y verificadas. Esta característica se integra estrechamente con el componente del

registro de contenedores que almacena las imágenes de contenedor.

- Implementa un **almacén cifrado** para poder guardar y recuperar de forma segura información secreta, como los nombres de usuario, las contraseñas y otros datos confidenciales.
- La plataforma se integra con la implementación existente del protocolo ligero de acceso a directorios (LDAP, por sus siglas en inglés) y de Active Directory (AD) para configurar un **control del acceso basado en funciones** (RBAC, por sus siglas en inglés) integrado.

Aqua, Prisma Cloud y StackRox ofrecen soluciones comerciales para proteger los contenedores y la infraestructura de Kubernetes. Snyk propone tanto soluciones comerciales como de código abierto.

Los proyectos de código abierto como Clair realizan análisis estáticos de vulnerabilidades en los contenedores de aplicaciones.

Falco, un proyecto incubado en la CNCF, es un supervisor de comportamientos diseñado para detectar anomalías en las aplicaciones nativas en la nube. Falco audita un sistema en su nivel más fundamental: el kernel. A continuación, contextualiza este dato con otras fuentes de información, como los parámetros del tiempo de ejecución del contenedor y los de Kubernetes, para avisar a los usuarios cuando se cumpla una de las reglas predefinidas.

Open Policy Agent (OPA), otro proyecto incubado en la CNCF, proporciona un conjunto de herramientas unificado y un marco para las políticas que se aplican en las distintas soluciones. OPA proporciona un lenguaje declarativo de alto nivel que permite a los desarrolladores y operadores especificar políticas como código y API sencillas para descargar del software la toma de decisiones relativas a políticas. OPA permite aplicar políticas en microservicios, Kubernetes, ciclos de CI/CD, puertas de enlace de API, etc.

Malla de servicios

La malla de servicios se está convirtiendo en un componente clave de la tecnología nativa en la nube. Permite controlar de forma detallada el tráfico que circula entre los distintos microservicios, al tiempo que proporciona información sobre el estado de cada uno de ellos.

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Aspen Mesh	F5	Consul	No presentado
Consul Enterprise	HashiCorp	Contour	En incubación
Grey Matter	Decipher Technology Studios	Envoy	Graduado
Kong Enterprise	Kong	Istio	No presentado
Linkerd Commercial Support	Buoyant	Kuma	Sandbox
Maesh	Containous	Linkerd	En incubación
Service Mesh Hub	Solo.io	Service Mesh Interface (SMI)	Sandbox
Tetrate	Tetrate	Zuul	No presentado

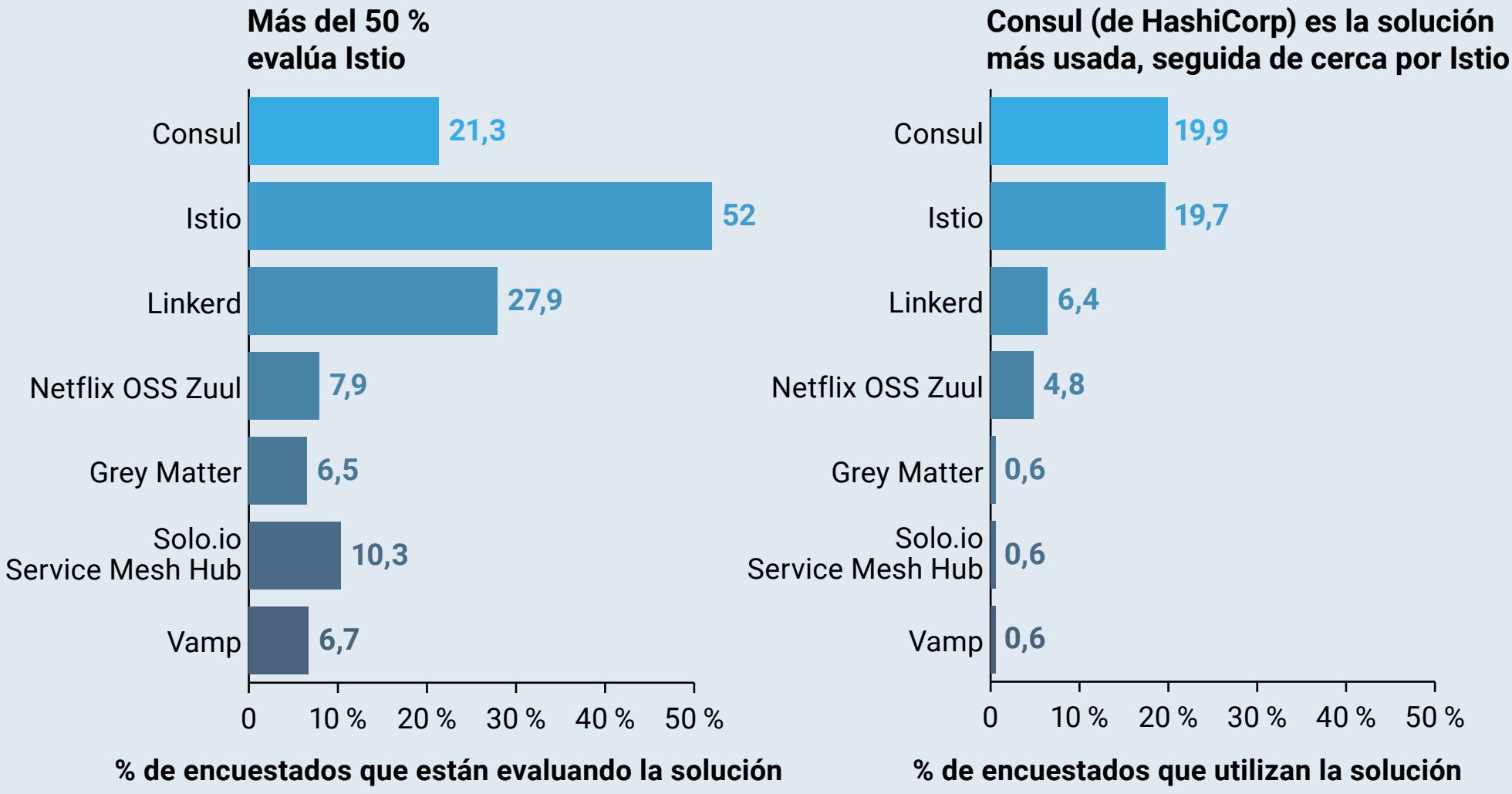
La malla de servicios complementa las redes nativas para contenedores. Se centra en el tráfico horizontal, mientras que la capa de la red principal se ocupa del vertical. La malla de servicios añade un proxy a cada microservicio, que intercepta el tráfico entrante y saliente. Al situarse cerca del microservicio, puede llevar un seguimiento del estado de dicho servicio.

Consul, Contour, Envoy, Istio, Kuma, Linkerd, Service Mesh Interface (SMI) y Zuul son algunos de los proyectos de malla de servicios de código abierto más conocidos.

Los clientes pueden elegir entre varias implementaciones comerciales, como Aspen Mesh (de F5), Consul Enterprise (de HashiCorp), Grey Matter (de Decipher Technology Studios), Kong Enterprise (de Kong), Linkerd Commercial Support (de Buoyant), Maesh (de Containous), Service Mesh Hub (de Solo.io) y Tetrate.

Volvamos una vez más a los datos de la encuesta de la CNCF de 2019 para fijarnos ahora en los relativos a la malla de servicios.

Según los resultados de la encuesta, las mallas de servicios planteaban dificultades al 27 % de los usuarios de Kubernetes, tanto en lo relativo a la elección de la herramienta más adecuada como a las operaciones cotidianas.



Fuente: Análisis de The New Stack de la encuesta de la CNCF realizada en 2019. P: Indique si su empresa/organización utiliza en el entorno de producción alguno de estos proyectos/productos de malla de servicios o está evaluando su adopción. Seleccione todas las respuestas que correspondan. ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan. Los datos solo se refieren a los encuestados que tienen al menos un clúster de Kubernetes (n=1082).

FIG. 2.5: *El hecho de que ya exista una base de desarrolladores que utilizan Consul para la prestación de servicios probablemente significa que también se está utilizando para funciones de malla de servicios relacionadas. Al mismo tiempo, más de la mitad de los usuarios de Kubernetes están considerando Istio por primera vez.*

Consul, de HashiCorp, e Istio resultaron ser las soluciones de malla de servicios más utilizadas entre los usuarios de Kubernetes, con cifras prácticamente iguales en ambos casos. Sin embargo, es posible que muchas de las personas que mencionaron Consul ya lo utilizasen para la detección de servicios y simplemente estuviesen probando su malla de servicios.

No era ese el caso de Istio y Linkerd, y ambas soluciones registraron un porcentaje de evaluación considerablemente mayor entre aquellos para quienes la malla de servicios resultaba complicada.

Especial mención merece la solución de Netflix Zuul, pues resultó ser la cuarta más evaluada y utilizada por los usuarios. Cabe destacar que solo el 23 % de los usuarios de Zuul se quejaban de dificultades relativas a la malla de servicios, en comparación con el 27 % del total de los encuestados. Dado que no gozaba de la misma notoriedad que Istio, es probable que los encuestados que utilizaban Zuul hubiesen conocido la solución por su capacidad de satisfacer una necesidad concreta.

Grey Matter, Service Mesh Hub y Vamp contaban con poquísimos usuarios, pero figuraban entre las soluciones evaluadas por ser bastante conocidas.

De cara al futuro, el principal reto al que tendrán que enfrentarse los usuarios de cualquier malla de servicios será su integración con las demás soluciones de Kubernetes de las que ya dispongan. Mientras tanto, seguramente Istio, Linkerd y otros servicios traten de mejorar su integración con Envoy.

Motor de orquestación de contenedores

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Canonical Distribution of Kubernetes (CDK)	Canonical	Kubernetes	Graduado
Cisco Container Platform	Cisco	OKD	No presentado
Docker Enterprise	Mirantis	Rancher Kubernetes Engine (RKE)	No presentado
HPE Ezmeral Container Platform	HPE	--	--
Mesosphere Kubernetes Engine (MKE)	D2iQ	--	--
Mirantis Cloud Platform	Mirantis	--	--
Rancher Kubernetes Engine (RKE)	Rancher Labs	--	--
Red Hat OpenShift	Red Hat	--	--
SUSE CaaS Platform	SUSE	--	--
Tanzu Kubernetes Grid	VMware	--	--

Como hemos visto en el primer capítulo, Kubernetes es el motor de orquestación de contenedores más habitual para las plataformas CaaS.

Aunque Kubernetes se puede implementar desde la distribución ascendente de código abierto disponible en GitHub, es posible que los clientes quieran invertir en una solución comercial que incluya paquetes de asistencia y mantenimiento. Canonical Distribution of Kubernetes, Cisco Container Platform, Docker Enterprise, HPE Ezmeral Container Platform, Mesosphere Kubernetes

Engine, Mirantis Cloud Platform, Rancher Kubernetes Engine, Red Hat OpenShift (que también se puede utilizar como PaaS), SUSE CaaS Platform y VMware Tanzu Kubernetes Grid son algunas de las propuestas comerciales disponibles en el mercado.

También existen proyectos de código abierto, como OKD y Rancher Kubernetes Engine (RKE).

Los clientes que hayan invertido en plataformas CaaS en la nube pública basadas en Google Kubernetes Engine o IBM Kubernetes Service pueden elegir Anthos o IBM Cloud Paks, que están perfectamente integrados con el plano de control que se ejecuta en la nube.

En el capítulo anterior se analizan a fondo los modelos de distribución de Kubernetes.

Registro de contenedores

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Docker Hub	Docker	**Docker Registry**	No presentado
Docker Trusted Registry	Mirantis	**Harbor**	Graduado
GitLab Container Registry	GitLab	**Project Quay**	No presentado
JFrog Container Registry / Artifactory	JFrog	**--**	--
Red Hat Quay	Red Hat	**--**	--

Los microservicios se empaquetan como imágenes de contenedor antes de que el motor de orquestación los implemente y amplíe. El registro de contenedores funciona como repositorio central donde se almacenan las imágenes de contenedor. Se integra con las herramientas de gestión de compilaciones para generar automáticamente imágenes cada vez que se cree una nueva versión del servicio. Una herramienta de gestión de versiones se encarga de sacar del registro la última versión de la imagen y de implementarla.

Docker Trusted Registry (de Mirantis), GitLab Container Registry, JFrog Container Registry y Red Hat Quay son algunos de los registros de contenedores disponibles en el mercado, que se suman a los que ofrecen los proveedores de servicios en la nube. Además, Docker ofrece un registro alojado llamado Docker Hub y su equivalente de código abierto, Docker Registry.

Además de almacenar las imágenes de contenedor, estos productos también suelen encargarse de la búsqueda de vulnerabilidades en las imágenes, la autenticación y la autorización de usuarios. Hay empresas que necesitan que estos repositorios estén presentes de forma local, así que prohíben el acceso a registros basados en Internet. Para esos casos, algunos de los productos comerciales disponen de registros internos. Si no se cuenta con una solución de este tipo, se puede crear un registro interno con muchas de las mismas funciones.

Los proveedores de servicios en la nube que ofrecen un servicio gestionado de Kubernetes brindan la posibilidad de alojar un registro de contenedores privado dentro de la cuenta del cliente. Como el clúster y el registro comparten la misma región y la misma cuenta, garantizan la seguridad y una baja latencia.

Harbor, inicialmente desarrollado por VMware, es un registro de imágenes de contenedor de código abierto que protege las imágenes mediante el control del acceso basado en funciones, busca vulnerabilidades en las imágenes y firma las imágenes para indicar que son de confianza. Project Quay, la versión de código abierto de Quay de Red Hat, ofrece una IU web pensada para el consumo personal, búsqueda de vulnerabilidades en imágenes y almacenamiento y protección de nivel empresarial.

Gestión de compilaciones

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
CircleCI	Circle Internet Services	GitLab	No presentado
CloudBees CI	CloudBees	Jenkins / Jenkins X	No presentado
Digital.ai	Digital.ai Software	--	--
GitHub Actions	GitHub	--	--
GitLab CI	GitLab	--	--
JFrog Pipelines	JFrog	--	--
SemaphoreCI	Rendered Text	--	--
Travis CI	Travis CI	--	--

Para que los microservicios se entreguen con rapidez, cada vez que llega código al repositorio, se crea una nueva imagen de contenedor y se envía al registro de contenedores. A continuación,

estas imágenes se implementan en entornos de prueba o de almacenamiento provisional dentro de los CaaS para realizar pruebas, ya sea de forma manual o automatizada.

La gestión de compilaciones pasa por convertir la versión más reciente del código en diversos artefactos, como imágenes de contenedor, bibliotecas y ejecutables. Constituye una fase importante del ciclo de integración/implementación continua (CI/CD).

El sistema de gestión de código abierto, basado en repositorios Git internos o externos, activa un proceso de compilación automatizado y seguro, que generará un artefacto de implementación. Esta fase puede dar lugar a imágenes de contenedor, charts de Helm e implementaciones de Kubernetes.

GitLab y Jenkins son productos de software de gestión de compilaciones muy conocidos que están disponibles tanto en versión comercial como de código abierto. CloudBees CI es una versión comercial del servidor de gestión de compilaciones de Jenkins.

CircleCI, Digital.ai, GitHub Actions, JFrog Pipelines, SemaphoreCI y Travis CI son soluciones de CI/CD comerciales para microservicios.

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
Armory Enterprise Spinnaker	Armory	Argo CD	En incubación
Open Enterprise Spinnaker	OpsMx	Flux	Sandbox

Gestión de versiones

La gestión de versiones constituye la última fase del ciclo de CI/CD y conlleva la implementación en el entorno de producción de una versión completamente probada de los microservicios. Forman parte de la gestión de versiones las estrategias de implementación avanzada como las versiones de valores controlados, las implementaciones azul/verde, las reversiones y las puestas al día.

Spinnaker es una de las herramientas de gestión de versiones que ha tenido mejor acogida para los microservicios. Automatiza la gestión y la implementación de artefactos, con lo que constituye un buen complemento para Jenkins y otras herramientas de gestión de compilaciones. Armory

y OpsMx cuentan con implementaciones comerciales de Spinnaker.

GitOps es una técnica emergente para implementar la configuración como código (CaC, por sus siglas en inglés). Los artefactos de YAML y los charts de Helm tienen control de versiones y se mantienen en un repositorio Git. Mediante una estrategia de inserción o extracción, los cambios realizados en la configuración se aplican en el clúster. Argo CD (proyecto incubado en la CNCF) y Flux (proyecto de sandbox de la CNCF) son dos proyectos de código abierto muy utilizados para configurar GitOps.

Productos comerciales	Proveedor	Proyectos de código abierto	Estado CNCF
AppDynamics	Cisco	Fluentd	Graduado
Datadog	Datadog	Grafana	No presentado
Dynatrace	Dynatrace	Jaeger	Graduado
Honeycomb	Honeycomb	OpenTelemetry	Sandbox
Lightstep	Lightstep	OpenTracing	En incubación
New Relic One	New Relic	Prometheus	Graduado
Sysdig Monitor	Sysdig	--	--

Observabilidad

La observabilidad pasa por capturar las métricas, los logs, los eventos y las trazas de todo el conjunto de soluciones.

La supervisión se encarga de recopilar métricas procedentes de las aplicaciones e infraestructuras de la plataforma. Una plataforma de supervisión sólida supervisa el estado del clúster de Kubernetes, así como las aplicaciones implementadas. Los agentes instalados en cada nodo recopilan información detallada sobre el uso de recursos, el estado de los clústeres y los nodos, la disponibilidad de almacenamiento y memoria, el número de contenedores que se ejecutan en cada nodo y métricas detalladas relacionadas con los pods. Los clientes pueden implementar herramientas de supervisión de código abierto como Grafana y Prometheus, o invertir en plataformas comerciales como DataDog, Dynatrace, New Relic One y Sysdig Monitor.

Mediante la creación de logs, se recopilan y agregan la información, los avisos y los errores

procedentes de varios componentes de la plataforma nativa en la nube. Casi todos los componentes de Kubernetes generan logs que proporcionan información detallada sobre el estado actual del clúster. Se recomienda a los desarrolladores que integren la creación de logs en los microservicios. Dentro del clúster, se implementan varios agentes para recopilar y enviar los logs a un repositorio central. Fluentd es una conocida herramienta de recopilación de datos de código abierto implementada en Kubernetes, que se puede utilizar junto con Elastic y Kibana para visualizar los logs y realizar búsquedas en ellos. Los programas de software de malla de servicios, como Istio y Linkerd, se integran a la perfección con las plataformas de creación de logs.

Tanto Prometheus como Fluentd son proyectos «graduados» de la CNCF.

Como las aplicaciones nativas en la nube se ensamblan a partir de servicios autónomos y dispares, es importante llevar un seguimiento de la cadena de comunicación y del tiempo que tarda en responder cada servicio. Este mecanismo resulta crucial para supervisar y analizar el rendimiento de las aplicaciones.

Las herramientas de código abierto como Jaeger y OpenTracing ofrecen funciones de supervisión del rendimiento de las aplicaciones (APM, por sus siglas en inglés) para los microservicios. Entre las propuestas comerciales, se encuentran AppDynamics, Honeycomb y Lightstep, las cuales brindan funciones de supervisión y rastreo integral para las aplicaciones modernas.

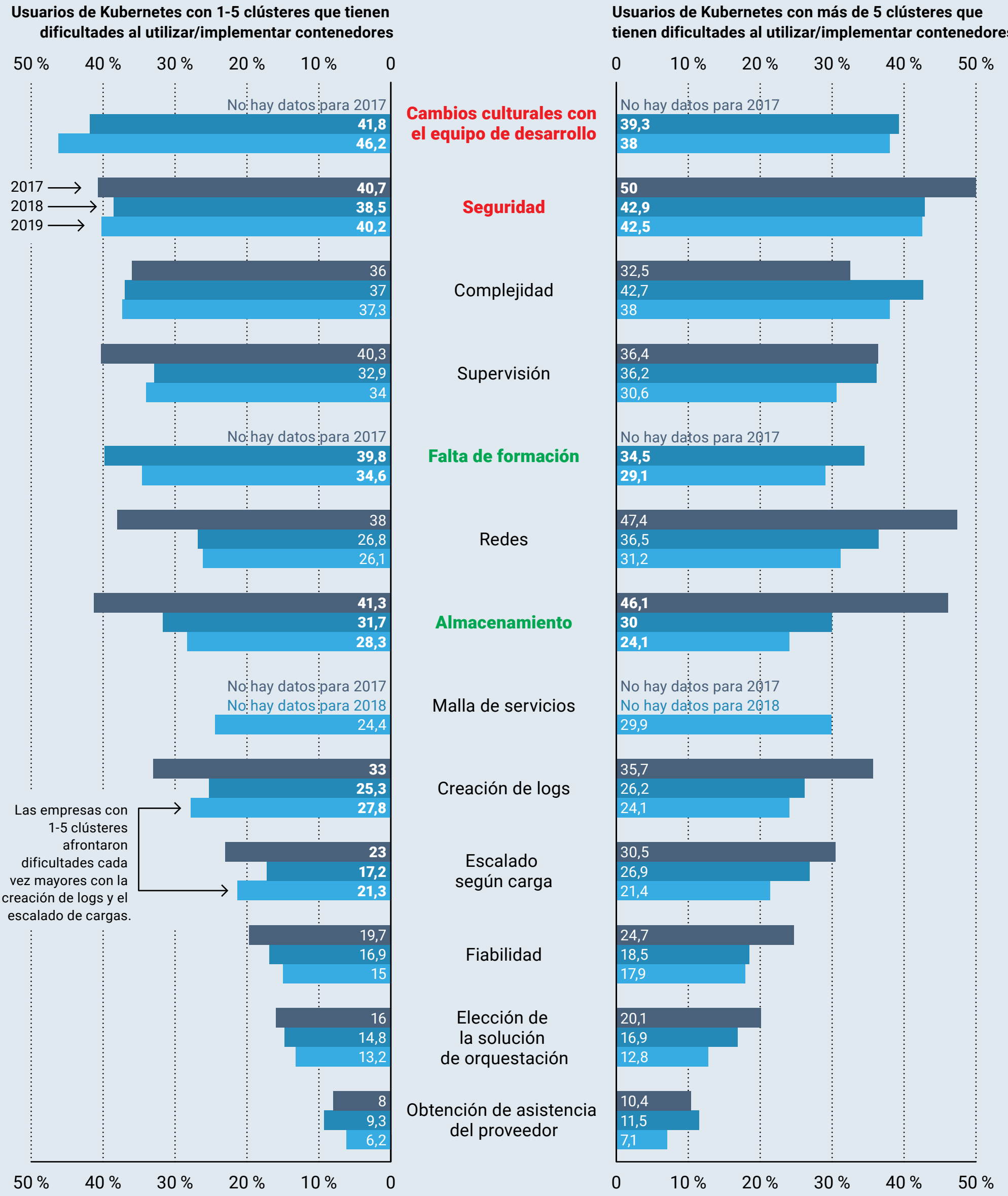
OpenTelemetry proporciona un solo conjunto de API, bibliotecas, agentes y servicios de recopilación para capturar trazas y métricas distribuidas procedentes de las aplicaciones nativas en la nube.

OpenTracing es un proyecto incubado en la CNCF, mientras que OpenTelemetry es un proyecto de sandbox.

Retos que plantean los contenedores a los usuarios de Kubernetes

Volvamos una vez más a los datos de las encuestas realizadas por la CNCF entre 2017 y 2019 para fijarnos en las dificultades que tienen los usuarios de Kubernetes con los contenedores.

La formación y el almacenamiento son más sencillos, pero la seguridad y el cambio cultural siguen planteando dificultades



Fuente: Análisis de The New Stack de las encuestas de la CNCF realizadas en 2017, 2018 y 2019. P: ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan. P: ¿Con qué dificultades se encuentran a la hora de utilizar/implementar contenedores? Seleccione todas las respuestas que correspondan.
1-5 clústeres de Kubernetes: 2017, n=300; 2018, n=925; 2019, n=729. >5 clústeres de Kubernetes: 2017, n=154; 2018, n=550; 2019, n=468.

FIG. 2.6: *Los usuarios de Kubernetes con 1-5 clústeres tienen dificultades con respecto al cambio cultural del equipo de desarrollo. Entre los que tienen más de cinco clústeres, todas las dificultades se redujeron ligeramente en 2019.*

En las empresas con más de cinco clústeres, las dificultades relacionadas con los contenedores mejoraron de forma considerable en 2018 y continuaron con la misma tendencia en 2019. Entre los usuarios de Kubernetes con 1-5 clústeres, las dificultades relacionadas con el uso o la implementación de contenedores mencionadas con más frecuencia fueron la seguridad, la complejidad y el cambio cultural que se genera en el equipo de desarrollo.

Independientemente del número de clústeres utilizados, según la CNCF, se estaban abordando los problemas de almacenamiento y de conectividad de datos. La reducción en los problemas de almacenamiento podría deberse a la madurez de la especificación CSI y a la disponibilidad de una interfaz CSI en la mayoría de las soluciones de almacenamiento. Tal vez las dificultades de conectividad hayan sido menos graves porque las tecnologías basadas en una interfaz de red de contenedores (CNI) como Flannel se convirtieron en el estándar para la comunicación de los contenedores dentro de los clústeres.

El éxito de la CNCF, KubeCon + CloudNativeCon y el resto del ecosistema de código abierto se pone de manifiesto en la constante reducción en la cantidad de usuarios que se quejan de falta de formación o de asistencia por parte de los proveedores.

Entre los encuestados con más de cinco clústeres, continuó la paulatina superación de las dificultades relacionadas con la ampliación de la implementación. Entre los casos de uso abordados por el ecosistema de Kubernetes, se encontraban la ampliación y la supervisión de cargas de trabajo y la creación de logs en clústeres más grandes.

Lo tenían peor las empresas con un máximo de cinco clústeres, donde en 2019 las dificultades aumentaron considerablemente en cuanto a la ampliación de implementaciones, la cultura del equipo de desarrollo y la creación de logs.

Resultaba especialmente problemático el cambio cultural en el equipo de desarrollo, porque las empresas más pequeñas no habían avanzado tanto en la reorganización de su estructura. Aunque muchas organizaciones formaron equipos de DevOps o ingenieros de fiabilidad de plataforma (SRE, por sus siglas en inglés), en estos entornos la función del desarrollador se sigue orientando a la gestión de aspectos relacionados con la capa de la infraestructura.

Plataformas de gestión de contenedores empresariales

Los proveedores de plataformas y de sistemas operativos han creado distribuciones comerciales de Kubernetes que se pueden instalar en un centro de datos empresarial. En lo que se refiere a la infraestructura, se puede elegir entre servidores físicos, máquinas virtuales ejecutadas en un hipervisor o infraestructura como servicio (IaaS) de nube pública.

A continuación enumeramos los principales proveedores que ofrecen una distribución comercial de Kubernetes:

Canonical

Ubuntu, una solución de Canonical, es el sistema operativo Linux en la nube pública más utilizado. Ubuntu también se ejecuta en nubes privadas y dispositivos de Internet de las cosas (IdC), además de como sistema operativo de ordenadores de sobremesa. Las organizaciones empresariales prefieren Ubuntu por su compatibilidad, rendimiento y versatilidad.

Canonical tiene dos versiones de Kubernetes:

- MicroK8s: una distribución ligera que se puede instalar en dispositivos IdC y perimetrales con un solo comando.
- Charmed Distribution of Kubernetes: versión ascendente de Kubernetes que garantiza la plena conformidad con la CNCF. Canonical ofrece un servicio gestionado para empresas, que incluye formación, instalación, configuración, herramientas de orquestación y optimización de los clústeres de Kubernetes en el entorno de producción.

Mirantis

Mirantis ha pasado de ser una empresa centrada en OpenStack a entrar en el campo de Kubernetes. De hecho, ahora la empresa cuenta con su propia distribución de Kubernetes. Se posiciona como empresa de infraestructuras de nube abiertas, con una plataforma capaz de gestionar con fluidez máquinas virtuales y contenedores.

En 2019, Mirantis compró Docker Enterprise a Docker, Inc., lo que incluye la edición Docker

Engine Enterprise, Docker Desktop Enterprise, Docker Trusted Registry y Docker Enterprise CaaS.

Además de Docker Enterprise, Mirantis también tiene una solución gestionada de Kubernetes llamada KaaS —Kubernetes como servicio— que se puede implementar en OpenStack o AWS.

Rancher

Rancher Labs (que en julio de 2020 estaba siendo adquirido por SUSE) proporciona una plataforma para implementar y gestionar Kubernetes como servicio empresarial.

Rancher Kubernetes Engine se orienta a los departamentos de TI empresariales que se estén planteando ofrecer a sus empleados Kubernetes como servicio. Además de mantener su propia distribución de Kubernetes, Rancher también gestiona otros productos alojados de contenedores como servicio —como Amazon Elastic Container Service for Kubernetes (Amazon EKS), Azure Kubernetes Service (AKS) y Google Kubernetes Engine (GKE)—, al tiempo que aplica políticas de seguridad coherentes. La plataforma se integra con Active Directory, LDAP y otros servicios de TI para proporcionar autenticación, control del acceso basado en funciones y políticas de seguridad en distintos clústeres de Kubernetes.

Red Hat

Red Hat OpenShift Container Platform es una plataforma de aplicaciones construida en Kubernetes. Se basa en tecnologías de código abierto de eficacia probada, como Red Hat Enterprise Linux, CRI-O y Kubernetes, para ofrecer una gestión integral del ciclo de vida de las aplicaciones.

Red Hat OpenShift Container Platform se puede implementar tanto en la nube pública como de forma local. La plataforma automatiza las compilaciones de aplicaciones y contenedores, las implementaciones, la ampliación y la gestión del estado de salud. Como distribución de Kubernetes certificada por la CNCF, OpenShift garantiza la portabilidad e interoperabilidad de las cargas de trabajo basadas en contenedores.

OpenShift se puede implementar en entornos de nube privada e híbrida, además de estar disponible como servicio gestionado en la nube pública. Red Hat ha unido fuerzas con proveedores públicos como Amazon, Microsoft y Google para ofrecer a los clientes una PaaS OpenShift gestionada.

SUSE

SUSE, la conocida empresa de distribución de Linux, se ha lanzado al mercado de los CaaS con una distribución de Kubernetes.

SUSE CaaS Platform permite implementar Kubernetes con varios nodos principales en entornos de nube pública y privada. Incluye características de seguridad empresarial como el control del acceso basado en funciones, el escaneo de imágenes y la compatibilidad con la seguridad de la capa de transporte (TLS). La plataforma también cuenta con un sistema operativo host de contenedores específico, tiempo de ejecución del contenedor y registro de imágenes de contenedor.

SUSE está aprovechando su presencia en las empresas para impulsar la adopción de la plataforma CaaS. Al igual que sus competidores, SUSE apuesta por llegar a ofrecer un conjunto de soluciones integradas basado en lo mejor de las tecnologías de código abierto.

VMware

Después de adquirir Pivotal Labs, VMware cambió el nombre de Pivotal Kubernetes Service (PKS), que pasó a llamarse VMware Tanzu Kubernetes Grid.

La distribución de Kubernetes que ofrece VMware está disponible en dos versiones: Tanzu Kubernetes Grid y Tanzu Kubernetes Grid Integrated Edition (TKGI).

Tanzu Kubernetes Grid es un tiempo de ejecución de Kubernetes adecuado para empresas y certificado por la CNCF que optimiza las operaciones en una infraestructura de varias nubes. Se puede implementar en vSphere (de forma local), IaaS (nube pública) o dispositivos con recursos limitados (perímetro/IdC).

Tanzu Kubernetes Grid Integrated Edition (TKGI) es una solución de contenedores basada en Kubernetes para entornos de producción dotada de conectividad avanzada, un registro de contenedores privado y gestión de todo el ciclo de vida. Está perfectamente integrada con vSphere, vCenter, vSAN y NSX. Se puede implementar tanto en un centro de datos donde se ejecuten las soluciones VMware como en la nube pública con VMware Cloud (VMC).

Plataformas de gestión de contenedores gestionadas

A continuación enumeramos los proveedores más conocidos que ofrecen Kubernetes como servicio gestionado:

Amazon Elastic Kubernetes Service

Amazon Elastic Kubernetes Service (EKS) es una solución de Kubernetes gestionada proporcionada por AWS. Se basa en los componentes clave de AWS, como Amazon Elastic Compute Cloud (EC2), Amazon EBS, Amazon Virtual Private Cloud (VPC) y Identity Access Management (IAM). Además, AWS también cuenta con un registro de contenedores integrado, Amazon Elastic Container Registry (ECR), que brinda un acceso seguro y de baja latencia a las imágenes de contenedor.

Amazon EKS se integra a la perfección con CloudWatch para la supervisión y con IAM para la seguridad. AWS App Mesh proporciona funciones de malla de servicios nativas para las cargas de trabajo implementadas en AWS. Mediante AWS Fargate, Amazon ha expuesto un mecanismo sin servidor para ejecutar contenedores en EKS.

Para acelerar la inferencia y el entrenamiento del aprendizaje automático, es posible programar cargas de trabajo en nodos conectados a unidades de procesamiento de gráficos (GPU) NVIDIA.

AWS Outposts, la plataforma de nube híbrida de Amazon, ejecuta EKS de forma local.

Azure Kubernetes Service

Azure Kubernetes Service (AKS) es una plataforma de gestión de contenedores gestionada que está disponible en Microsoft Azure. Se construye sobre máquinas virtuales Azure, Azure Storage, Virtual Networking y Azure Monitoring. Azure Container Registry (ACR) se puede aprovisionar en el mismo grupo de recursos del clúster AKS para ofrecer acceso privado a las imágenes de contenedor.

AKS recurre a los conjuntos de escalado de máquinas virtuales y los conjuntos de disponibilidad para aumentar y reducir de forma dinámica el número de nodos de un clúster.

AKS está disponible en Azure Stack Hub, la solución de nube híbrida de Microsoft para ejecutar servicios Azure en centros de datos.

Google Kubernetes Engine

Google Kubernetes Engine (GKE) fue uno de los primeros servicios gestionados en ofrecer Kubernetes en la nube pública. Al igual que otros CaaS basados en la nube pública, GKE utiliza los servicios principales de Google Cloud Platform, como Compute Engine, Persistent Disks, VPC y Stackdriver.

Google ha hecho posible la disponibilidad de Kubernetes en entornos locales y otras plataformas de nube pública mediante el servicio Anthos, un plano de control que se ejecuta en GCP pero gestiona todo el ciclo de vida de los clústeres lanzados en entornos híbridos y de varias nubes.

Google ha ampliado GKE con un servicio Istio gestionado para ofrecer funciones de malla de servicios. Ofrece, además, Cloud Run, una plataforma sin servidor —basada en el proyecto de código abierto Knative— para ejecutar contenedores sin lanzar clústeres.

IBM Cloud Kubernetes Service

IBM Cloud Kubernetes Service (IKS) es una solución gestionada para crear clústeres de Kubernetes de hosts de computación con el fin de implementar y gestionar aplicaciones basadas en contenedores en IBM Cloud. En calidad de proveedor certificado, IKS proporciona programación inteligente, autorreparación, escala horizontal, detección de servicios y equilibrio de cargas, implantaciones y reversiones automatizadas, y gestión de configuraciones e información secreta para las aplicaciones modernas.

IBM es uno de los pocos proveedores de servicios en la nube que ofrecen un servicio Kubernetes gestionado en versión física.

Gracias a la adquisición de Red Hat, IBM ofrece diversos clústeres de OpenShift o Kubernetes disponibles mediante IKS.

IBM también ha creado Cloud Paks, un conjunto de dispositivos hiperconvergentes basados en Kubernetes y OpenShift que se pueden ejecutar en centros de datos empresariales.

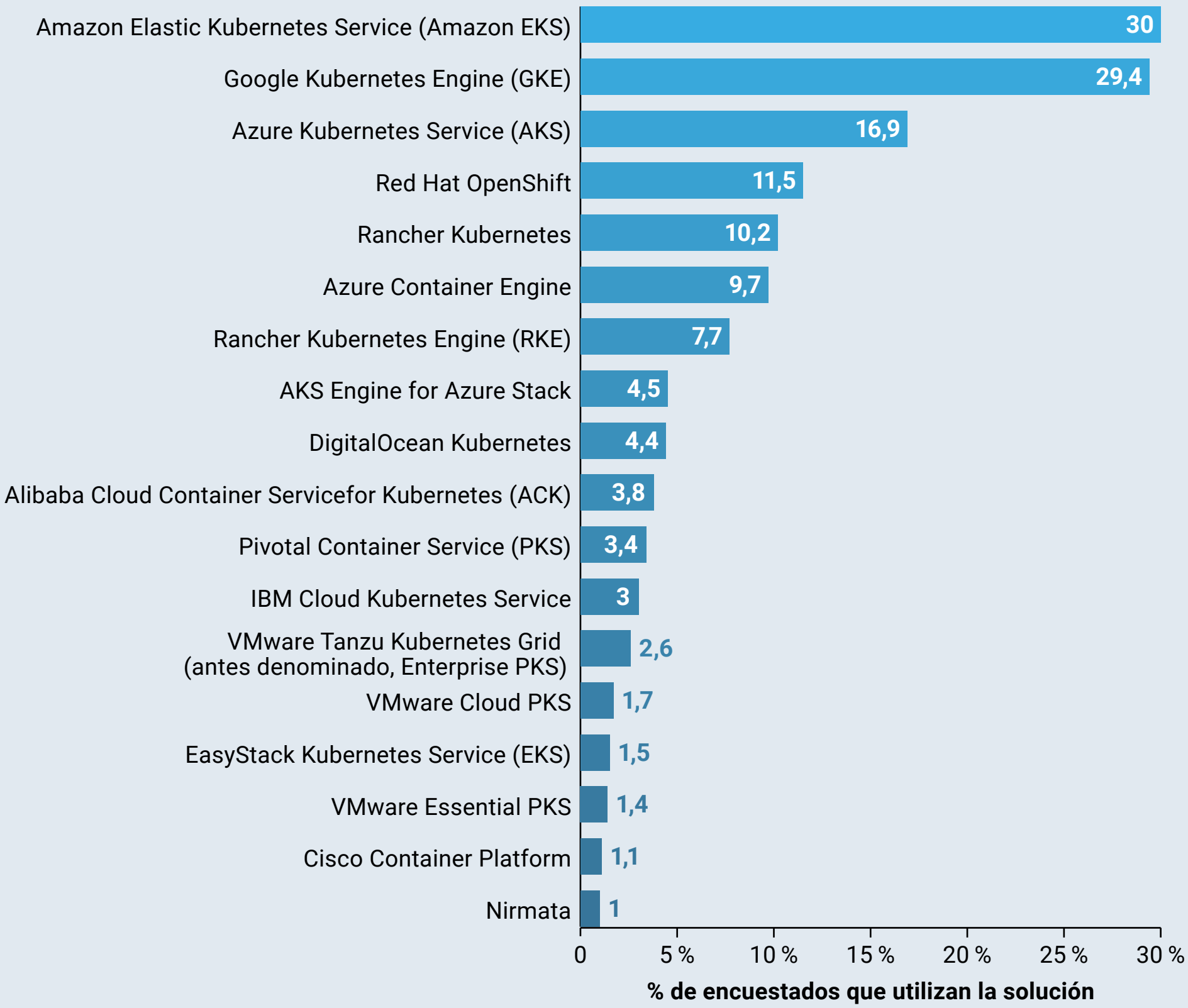


FIG. 2.7: *Desde que se formuló la pregunta, han cambiado los nombres de numerosos productos. Junto con los productos de Pivotal y VMware, es probable que Tanzu goce de un puesto más elevado cuando se publiquen los resultados de la encuesta de 2020.*

Otros servicios gestionados de Kubernetes

Además de los servicios enumerados, los clientes también tienen a su disposición los que ofrecen Alibaba Cloud Container Service, DigitalOcean Kubernetes, Huawei Cloud Container Engine y Platform9.

Adopción de Kubernetes como servicio gestionado por proveedores de nube

Cuando la CNCF realizó su encuesta en 2017, el 25 % de los usuarios de Kubernetes encuestados ejecutaban contenedores con Google Kubernetes Engine (GKE). AWS aún estaba promocionando su Elastic Container Service (ECS) y Microsoft acababa de lanzar Azure Container Service. Pero ya entonces estaba claro que Kubernetes se acabaría convirtiendo en el orquestador de contenedores por excelencia. Aunque el mercado de Kubernetes estaba dando sus primeros pasos, el 52 % de los usuarios de Kubernetes lo ejecutaban en un entorno de AWS y el 34 % en Google Cloud Platform (GCP), mientras que Azure y VMware registraban un 21 % y un 15 %, respectivamente. La mitad de las cargas de trabajo de Kubernetes se ejecutaban en AWS aunque se ejecutasen sobre MV y estuviesen gestionadas por clientes de AWS.

En el AWS re:Invent de ese año, se presentó Amazon Elastic Container Service for Kubernetes (EKS), que estaría disponible de forma generalizada a mediados de 2018, y AWS pasó a ser miembro de la CNCF poco después. Kubernetes había ganado la guerra de la orquestación de contenedores, pero las empresas tendían a ejecutar los contenedores donde ejecutaban otras cargas de trabajo. Por este motivo, la adopción de Amazon EKS y Azure Kubernetes Service (AKS) fue considerable en 2019.

La última encuesta de la CNCF, realizada en 2019, incluía más de 100 opciones en la pregunta sobre el modo de gestionar los contenedores, y se observó que el uso de Amazon EKS (30 %) era un poco más frecuente que el de GKE (29 %) entre los usuarios de Kubernetes. Además de los productos de Microsoft, también Red Hat OpenShift y Rancher Kubernetes alcanzaron cifras importantes.

PRISMA
BY PALO ALTO NETWORKS

Kubernetes ha evolucionado, y su seguridad no debería quedarse atrás

Kubernetes incorpora muchas funciones de seguridad, pero eso no significa que sea seguro por defecto. La gestión de dependencias sigue sin ser segura y siempre surgen nuevos vectores de ataques, como los contenedores maliciosos.
A pesar de todos los avances en materia de seguridad, la API sigue siendo el principal punto de entrada a Kubernetes para los atacantes.

La buena noticia es que, en los últimos años, los equipos de seguridad han aprendido mucho sobre cómo proteger las implementaciones de Kubernetes y las aplicaciones que se ejecutan en los contenedores. Defenderse de las amenazas es más fácil gracias a una combinación de procesos y herramientas dirigidas a desarrolladores, equipos de Seguridad y operaciones de TI (DevSecOps).
Por ejemplo, las herramientas de detección y análisis de anomalías permiten identificar fácilmente los contenedores maliciosos y demás vectores de ataque.

Robert Haynes, evangelista de la seguridad en la nube de Prisma Cloud de Palo Alto Networks, habla de la seguridad de Kubernetes más allá de sus funciones nativas. También hace un repaso de la evolución del panorama de las vulnerabilidades de Kubernetes desde que, hace unos años, se produjeran los primeros ataques de API. Según, Haynes, los primeros ataques cruciales «nos enseñaron algo que ya sabíamos: que Kubernetes sigue siendo un software y que todo software tiene vulnerabilidades».

Escuchar en SoundCloud

Robert Haynes *pasó de ser un humilde técnico del servicio de asistencia a todo un administrador de sistemas de Unix para después regresar al marketing. Actualmente, forma parte del equipo de Prisma Cloud de Palo Alto Networks, donde disfruta contando historias sobre la seguridad, las tecnologías y las personas.*

Cómo combatir el exceso de complejidad de Kubernetes

Cada vez son más las organizaciones que se pasan a Kubernetes y otros entornos nativos en la nube a gran escala, por lo que gestionar clústeres resulta cada vez más complejo. Este exceso de complejidad de Kubernetes se ha convertido en un verdadero problema.

En este podcast, Ravi Lachhman, evangelista de Harness, y Frank Moley, responsable sénior de Ingeniería Técnica de DataStax, hablan sobre este problema y sobre qué se puede hacer para paliarlo.

En parte, este exceso de complejidad es atribuible a que cuando se produce el cambio de paradigma, los equipos de DevOps se ven obligados a ampliar sus áreas de especialidad, a menudo fuera de su zona de confort. Los miembros del equipo de Infraestructura, por ejemplo, tienen que estar al tanto de las necesidades de los desarrolladores, mientras que las cargas de trabajo de los desarrolladores cubren, cada vez más, tareas relacionadas con la infraestructura.

Ayudar a los equipos de DevOps a reducir el exceso de complejidad es cuestión de reconocer las propias limitaciones. Para conseguir nuestros objetivos, Lachhman nos recomienda comprar los componentes poco a poco y encontrar la mejor forma de gestionar las deficiencias.

Escuchar en SoundCloud

Ravi Lachhman *es evangelista de Harness. Antes de trabajar para Harness, Lachhman fue evangelista de AppDynamics y ostentó varios cargos en las áreas de ventas e ingeniería en Mesosphere, Red Hat e IBM.*

Frank Moley *es responsable sénior de Ingeniería Técnica de DataStax y autor de contenido de LinkedIn Learning. Está especializado en informática nativa en la nube, arquitecturas de microservicios, ingeniería de plataformas y seguridad.*

Kelsey Hightower nos habla sobre su recorrido personal con Kubernetes

En este podcast, hablamos con Kelsey Hightower, toda una eminencia en el ecosistema de Kubernetes, Developer Advocate principal para Google Cloud y antiguo miembro del comité de supervisión técnica de la Cloud Native Computing Foundation (CNCF) que regula todos los proyectos de la entidad, incluido Kubernetes. Hightower nos explica su papel en el desarrollo de Kubernetes y los retos que tiene por delante.

Sobre su primer contacto con Kubernetes, allá por 2015, Hightower recuerda: «Cuando me puse a examinar la base de código, no había ninguna reseña buena sobre cómo instalarlo. Así que lo primero que hice fue crear lo que podrían considerarse las primeras guías de instalación».

Actualmente, Hightower quiere que la curva de aprendizaje se allane conforme la plataforma continúa madurando. Y añade: «No tiene ningún sentido que los desarrolladores tengan que ser expertos en la red de Kubernetes para poder desempeñar su trabajo como ingenieros de software».

Escuchar en SoundCloud

Kelsey Hightower es Developer Advocate principal para Google Cloud. Colaboró en el desarrollo y la mejora de muchos de los productos de Google Cloud, como Google Kubernetes Engine, Google Cloud Functions y la API Gateway de Apigees.

Conclusión

Han pasado poco más de cinco años desde el lanzamiento de Kubernetes y hoy ya ha alcanzado un buen nivel de madurez. Los conceptos de «pods», «nodos» y «clústeres» forman parte de un vocabulario que describe las arquitecturas con escalabilidad horizontal. Ya no se trata tanto de las maravillas del sistema interno como de lo que se construye en Kubernetes.

Kubernetes se está volviendo aburrido, simplemente porque ahora es resiliente y escalable. Sencillamente, el sistema funciona.

Resulta fácil olvidar que, hace unos años, el sistema interno de Kubernetes era de todo menos aburrido. Pero desde que se publicó el primer libro electrónico de The New Stack sobre Kubernetes en 2017, las continuas actualizaciones del orquestador de contenedores han dado lugar a un proceso iterativo que permite consolidar la arquitectura subyacente para las aplicaciones distribuidas con escalabilidad horizontal.

Ahora el reto consiste en cómo construir con Kubernetes como base. En el mundo de las arquitecturas con escalabilidad horizontal, lo que se distribuye en contenedores es el código. El código es el componente del contenedor. El contenedor es un proceso que se ejecuta en un sistema operativo. Y, como proceso, es portátil. Se ha diseñado para ser inmutable, lo que significa que los contenedores se pueden sustituir por otros nuevos para actualizar el código del componente de la aplicación. La infraestructura inmutable se basa en la idea de que una aplicación se puede implementar en un entorno homogéneo y, en consecuencia, se actualiza constantemente de forma fluida sustituyendo los contenedores con el código actualizado.

La dificultad de construir sobre la base de Kubernetes se debe a la naturaleza de los contenedores como procesos inmutables. Un contenedor carece de estado. Para lograr que Kubernetes resulte útil para las empresas, los ingenieros han tenido que desarrollar interfaces que permiten conectar los contenedores sin estado.

La interfaz de red de contenedores (CNI) conecta los contenedores y sus correspondientes complementos con la red que utiliza la organización para ejecutar una aplicación. En este entorno, lo interesante es la amplia compatibilidad, que permite conectar complementos

en las redes. Las empresas que desarrollan una CNI común nos dan una pista sobre por qué la comunidad de Kubernetes en su conjunto tiene tanto éxito. Se debe al espíritu de cooperación que reina en la comunidad del código abierto, y que queda patente en los proyectos de Kubernetes. Tal vez hoy el sistema interno de Kubernetes resulte aburrido, pero los tecnólogos que han desarrollado la arquitectura subyacente derrochan energía y pasión por su trabajo, y construyen una arquitectura para uno de los primeros sistemas operativos para software y servicios en una infraestructura distribuida avanzada.

Sin embargo, Kubernetes aún es relativamente nuevo. Los sistemas operativos basados en cliente se crean para ordenadores personales o servidores, principalmente para aplicaciones monolíticas. Linux domina el ámbito de los servidores. Los contenedores se basan en Linux, son una extensión del kernel de Linux. Con Kubernetes, el contenedor se convierte en un proceso básico para orquestar el código que se ejecuta como componentes. Es un plano de control independiente de las arquitecturas en la nube o de cliente. Permite a las API gestionar el plano de control y la computación, las redes y el almacenamiento correspondientes.

La corta edad de Kubernetes se refleja en la experiencia de los desarrolladores. La arquitectura de Kubernetes aún no está del todo lista para los desarrolladores. La era de las plataformas como servicio (PaaS) está quedando atrás, ya tiene mucha menos relevancia que hace tres años. La PaaS brinda un modo de ejecutar aplicaciones en servicios en la nube y entornos empresariales. Se construye y ensambla como una estructura de aplicaciones, pero ni la mejor PaaS está a la altura de todos los casos extremos, principalmente las aplicaciones monolíticas más antiguas.

Los contenedores están cambiando las reglas del juego: ahora, se puede ejecutar código en un contenedor y gestionarlo con mucha más facilidad en mucho menos tiempo. Pero todavía hay pocas formas de desarrollar servicios en Kubernetes. Aquí entran en juego los contenedores como servicio (CaaS), también conocidos como plataformas de gestión de contenedores, que integran varios servicios nativos en la nube para su ejecución tanto en arquitecturas de microservicios modernas como en empresas monolíticas. Estas plataformas de gestión ejecutan varios servicios en Kubernetes. Empiezan por la CPU y la infraestructura física. La plataforma de gestión abarca el sistema operativo optimizado para contenedores, el tiempo de ejecución de los contenedores, las redes nativas para contenedores, el almacenamiento, la seguridad, el registro, la observabilidad,

el propio motor de orquestación de contenedores, etc.

Las plataformas CaaS tardarán un tiempo en alcanzar su madurez. Los servicios gestionados estarán cada vez más presentes y ahorrarán trabajo a los desarrolladores.

Hasta entonces, los papeles que desempeñan las personas —los roles, por así decirlo— evolucionarán en el caso de los desarrolladores, los encargados de configurar los servicios y los responsables de definir las políticas de Kubernetes. El universo de los contenedores aporta una nueva visión de lo que significa un sistema operativo cuando se ejecuta en la infraestructura, independientemente de dónde se ejecute.

Bien pensado, puede que Kubernetes no sea tan aburrido. Pero acabará siéndolo. Como suele decir Kelsey Hightower, lo próximo será hacerlo desaparecer.

Aviso

Además de los patrocinadores del libro electrónico, las siguientes empresas que en él se mencionan son patrocinadoras de The New Stack:

Accurics, AppDynamics, Aspen Mesh, Amazon Web Services, CircleCI, Citrix, CloudBees, Cloud Foundry Foundation, Cockroach Labs, Dell Technologies, Diamanti, Futurewei, GitLab, HAProxy, HashiCorp, Honeycomb, InfluxData, Lightbend, Lightstep, LogDNA, MayaData, MongoDB, New Relic, NS1, Packet, PagerDuty, Portworx, Red Hat, Redis Labs, Rezilion, SaltStack, SAP, Sentry, Snyk, Sonatype, The Linux Foundation, TriggerMesh y VMware.