Kubernetes: sternik, który przypłynął z Google

Poziom: podstawy **Dotyczy: Kubernetes 1.30, k3s 1.30 (wariant lekki)**

Czas: ~12 min

Ta strona otwiera książkę o Kubernetesie: skąd wziął się system, który zarządza kontenerami w tysiącach firm na świecie, dlaczego Zielona trzyma go tylko w laboratorium, i co czeka na kolejnych stronach.

Skróty użyte na tej stronie (4)

Skrót	Co znaczy
CNCF	Cloud Native Computing Foundation: fundacja pod Linux Foundation, formalny właściciel projektu Kubernetes od lipca 2015 roku.
CRI	Container Runtime Interface: standard, którym Kubernetes rozmawia z silnikiem kontenerów
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
K8s	Kubernetes: potoczny skrót nazwy, litera K, cyfra 8 oznaczająca liczbę liter pominiętych między K i s, litera s.

Skąd się wzięło

Zanim powstał Kubernetes, Google od lat uruchamiał własne aplikacje na systemie o nazwie Borg. Borg działał wewnątrz firmy od około 2003 roku i zarządzał niemal wszystkimi usługami Google, od wyszukiwarki po Gmaila, uruchamiając na wspólnej puli maszyn setki tysięcy zadań naraz. System pozostawał tajemnicą firmową przez ponad dekadę: świat zewnętrzny znał tylko plotki o skali, w jakiej Google układa swoje kontenery.

Następcą Borga był Omega, projekt eksperymentalny z bardziej elastycznym planowaniem zadań, który pozwalał wielu komponentom jednocześnie decydować, gdzie coś uruchomić, zamiast trzymać się jednego centralnego planisty. Omega nigdy nie zastąpiła Borga w pełni, ale wypracowane w niej pomysły trafiły później do trzeciego systemu, tym razem już otwartego dla świata.

W czerwcu 2014 roku Google ogłosiło Kubernetesa: projekt open source napisany od zera w języku Go przez inżynierów Craiga McLuckiego, Joe Bedę i Brendana Burnsa, korzystający z doświadczeń Borga i Omegi, ale bez ani linijki ich zamkniętego kodu. Nazwa pochodzi z greki i oznacza sternika albo pilota okrętu, stąd koło sterowe w logo projektu. W codziennym użyciu przyjął się skrót K8s: litera K, cyfra 8 oznaczająca liczbę liter pominiętych między K i s w słowie „kubernetes", i litera s.

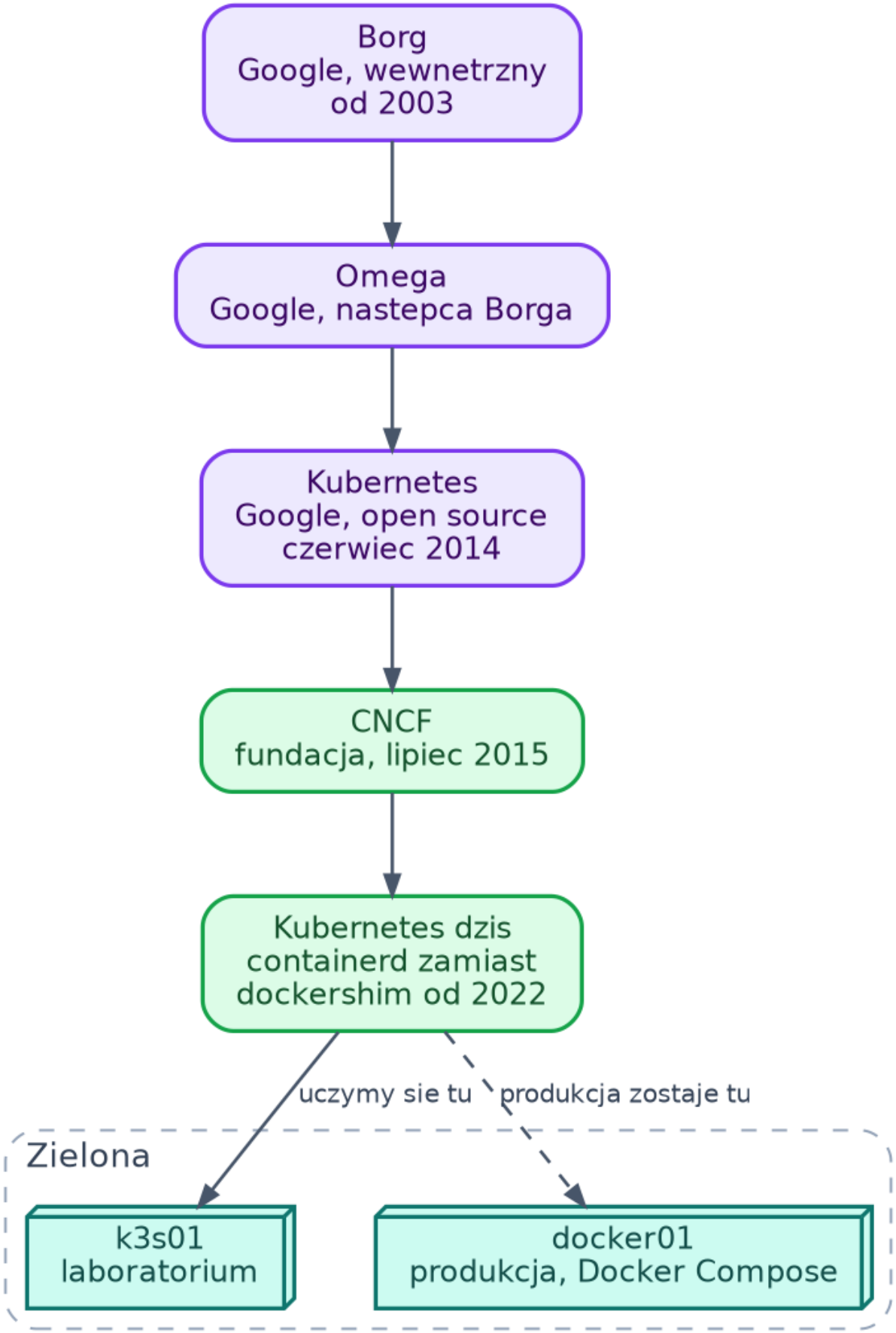
> **Pojęcie: orkiestracja kontenerów.** Automatyczne rozmieszczanie, uruchamianie i pilnowanie kontenerów na wielu maszynach naraz, tak żeby administrator opisywał tylko pożądany stan (ile kopii, ile zasobów), a system sam decydował, na którym węźle coś uruchomić i co zrobić, gdy węzeł odmówi posłuszeństwa.

Rok później, w lipcu 2015 roku, Google przekazało Kubernetesa nowo powstałej Cloud Native Computing Foundation, w skrócie CNCF, działającej pod skrzydłami Linux Foundation. Oddanie projektu fundacji neutralnej wobec dostawców oznaczało, że żadna pojedyncza firma, nawet Google, nie decyduje już samodzielnie o jego kierunku rozwoju. Dziś pod CNCF działają dziesiątki powiązanych projektów, ale Kubernetes pozostaje tym pierwszym i najbardziej rozpoznawalnym.

Przez pierwsze lata Kubernetes uruchamiał kontenery za pośrednictwem Dockera, korzystając z dodatkowej warstwy tłumaczącej zwanej dockershim. W wersji 1.20, ogłoszonej w grudniu 2020 roku, projekt zapowiedział wycofanie tej warstwy na rzecz bezpośredniej rozmowy z silnikami zgodnymi ze standardem CRI, takimi jak containerd. Zapowiedź wywołała falę niepokoju w internecie, bo wielu administratorów czytało ją jako koniec obsługi obrazów Dockera, choć w rzeczywistości zmieniał się tylko silnik uruchamiający te same obrazy. Dockershim zniknął ostatecznie w wersji 1.24, wydanej w kwietniu 2022 roku.

Jak to działa w jednym obrazku

> **Pojęcie: CNCF.** Cloud Native Computing Foundation, fundacja non profit pod Linux Foundation, która od 2015 roku jest formalnym właścicielem projektu Kubernetes i dba o to, by rozwój pozostał otwarty i niezależny od jednego dostawcy chmury.


Borg
Google, wewnetrzny
od 2003
Omega
Google, nastepca Borga
Kubernetes
Google, open source
czerwiec 2014
CNCF
fundacja, lipiec 2015
Kubernetes dzis
containerd zamiast
dockershim od 2022
uczymy sie tu
produkcja zostaje tu
Zielona
k3s01
laboratorium
docker01
produkcja, Docker Compose

Od Borga przez Omegę do Kubernetesa: rodowód orkiestracji kontenerów w Google i miejsce klastrów w Zielonej (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> Podczas przeglądu architektury na 2024 rok Tomek proponuje przeniesienie MailCow i kilku innych kontenerów z `docker01` na klaster Kubernetes. Po dwóch dniach lektury dokumentacji stawia sobie proste pytanie: kto w dwuosobowym zespole będzie w środku nocy diagnozował awarię `etcd`, kiedy jedyny klaster produkcyjny przestanie odpowiadać? Odpowiedzi nie ma, więc plan zamienia się w laboratorium.

Zespół Zielonej trzyma Kubernetesa wyłącznie na `k3s01`, lekkiej dystrybucji opisanej bliżej na stronie k3s w laboratorium: pierwszy deploy, nginx w kolorach, gdzie na przykładzie prostego, kolorowego nginksa uczy się Traefika i podstawowych wzorców wdrożeń bez ryzyka dla produkcji. Produkcja zostaje na `docker01`, uruchamiana przez zwykły Docker Compose, opisany w książce o Dockerze.

Powód jest uczciwy, nie ideologiczny: Kubernetes rozwiązuje problemy skali i samonaprawy, których dwuosobowy zespół IT z kilkunastoma kontenerami po prostu nie ma. Dodatkowa warstwa `etcd`, kontrolerów i sieci nakładek kosztowałaby więcej czasu na utrzymanie, niż oszczędza na automatyzacji, co strona Po co Kubernetes, skoro mamy Dockera nazywa wprost przerostem formy nad treścią. Gdyby Zielona kiedyś urosła do skali wymagającej setek kontenerów rozproszonych na wielu maszynach, doświadczenie z `k3s01` będzie gotowym punktem startu zamiast nauki od zera pod presją.

Mapa tej książki

Rozdział	Na jakie pytanie odpowiada
r1. Fundamenty i architektura	Czym jest Kubernetes, z czego się składa klaster i jak opisujemy stan w manifestach
r2. Obiekty warstwy aplikacji	Jak uruchomić i aktualizować aplikację za pomocą poda, deploymentu i innych kontrolerów
r3. Sieć w klastrze	Jak ruch trafia do poda, jak działa DNS i jak wpuszczamy ruch z zewnątrz
r4. Konfiguracja i magazyn	Jak przekazać aplikacji ustawienia, sekrety i trwałą przestrzeń dyskową

Rozdział	Na jakie pytanie odpowiada
r5. Bezpieczeństwo	Kto w klastrze może co robić i jak to ograniczać przez RBAC
r6. Diagnostyka i utrzymanie	Co robić, gdy coś w klastrze nie działa, i jak go bezpiecznie aktualizować
r7. Laboratorium k3s	Jak wygląda cała ścieżka na żywym przykładzie: kolorowy nginx i Traefik
r8. Dalej w laboratorium	Jak spakować kolorowy nginx w wykres Helm, skalować go automatycznie, obserwować metrykami i alertami oraz wdrażać z Gitea przez Argo CD

Powiązane: Po co Kubernetes, skoro mamy Dockera · k3s w laboratorium: pierwszy deploy, nginx w kolorach

Źródła: kubernetes.io/blog/2015/07/cloud-native-computing-foundation, kubernetes.io/blog/2020/12/02/dockershim-faq, kubernetes.io/blog/2022/02/17/dockershim-faq, en.wikipedia.org/wiki/Kubernetes, en.wikipedia.org/wiki/Borg_(cluster_manager)

Revision #23
Created 2026-09-28 17:32:52 UTC by Tomasz Konieczny
Updated 2026-10-26 14:16:49 UTC by Tomasz Konieczny