

# Spis treści

<i>Wstęp</i>	11
<i>Podziękowania</i>	12
<i>O książce</i>	13

## **TYDZIEŃ I. SZYBKA DROGA DO OPANOWANIA KUBERNETESA . . . . 19**

1.	<i>Zanim zaczniesz</i>	21
1.1.	Jak działa Kubernetes?	22
1.2.	Czy ta książka jest dla Ciebie?	26
1.3.	Tworzenie środowiska laboratoryjnego	27
1.4.	Natychmiastowa efektywność	33
2.	<i>Uruchamianie kontenerów w Kubernetesie za pomocą kapsuł i wdrożeń</i>	34
2.1.	Jak Kubernetes uruchamia kontenery i zarządza nimi?	35
2.2.	Uruchamianie kapsuł za pomocą kontrolerów	41
2.3.	Definiowanie wdrożeń w manifestach aplikacji	48
2.4.	Praca z aplikacjami działającymi w kapsułach	52
2.5.	Zarządzanie zasobami przez Kubernetes	56
2.6.	Laboratorium	59
	<i>Łączenie kapsuł przez sieć za pomocą usług</i>	60
3.1.	Jak Kubernetes routuje ruch sieciowy?	61
3.2.	Routowanie ruchu między kapsułami	65
3.3.	Routowanie do kapsuł ruchu zewnętrznego	70
3.4.	Routowanie ruchu poza Kubernetes	75
3.5.	Jak działa rozwiązywanie usług w Kubernetesie?	81
3.6.	Laboratorium	86
4-	<i>Konfigurowanie aplikacji za pomocą obiektów ConfigMap i Secret</i>	87
4.1.	Jak Kubernetes dostarcza konfigurację do aplikacji?	88
4.2.	Zapisywanie plików konfiguracyjnych w obiektach ConfigMap oraz ich używanie	93

4.3.	Udostępnianie danych konfiguracyjnych z obiektów ConfigMap	98
4.4.	Konfigurowanie poufnych danych za pomocą obiektów Secret	106
4.5.	Zarządzanie konfiguracją aplikacji w Kubernetesie	113
4.6.	Laboratorium	115
5.	<i>Przechowywanie danych przy wżyciu woluminów, punktów montowania i żądań</i>	<i>117</i>
5.1.	Jak Kubernetes buduje system plików kontenera?	118
5.2.	Przechowywanie danych na węźle za pomocą woluminów i punktów montowania	123
5.3.	Użycie woluminów trwałych oraz żądań do przechowywania danych dla całego klastra	131
5.4.	Dynamiczna alokacja woluminów i klasy pamięci masowej	141
5.5.	Opcje wyboru pamięci masowej w Kubernetesie	147
5.6.	Laboratorium	148
6.	<i>Używanie kontrolerów do skalowania aplikacji w celu rozproszenia ich na wiele kapsuł</i>	<i>149</i>
6.1.	Jak Kubernetes uruchamia skalowalne aplikacje?	150
6.2.	Używanie wdrożeń i zbiorów replik do skalowania pod kątem obciążenia	156
6.3.	Używanie kontrolerów DaemonSet do skalowania pod kątem zapewniania wysokiej dostępności	165
6.4.	Własność obiektów w Kubernetesie	171
6.5.	Laboratorium	174
<b>TYDZIEŃ II. KUBERNETES W PRAWDZTWYM ŚWIECIE</b>		<b>175</b>
7.	<i>Rozszerzanie aplikacji o wielokontenerowe kapsuły</i>	<i>177</i>
7.1.	Jak kontenery komunikują się w kapsule?	178
7.2.	Konfigurowanie aplikacji za pomocą kontenerów inicjujących	184
7.3.	Zapewnianie spójności za pomocą kontenerów adapterów	191
7.4.	Tworzenie warstwy abstrakcji połączeń za pomocą kontenerów ambasadorów	195
7.5.	Środowisko kapsuły	199
7.6.	Laboratorium	204

8. *Wykorzystywanie kontrolerów StatefulSet i Job do uruchamiania aplikacji operujących na dużych ilościach danych* 205
  - 8.1. Jak Kubernetes modeluje stabilność za pomocą kontrolerów StatefulSet 206
  - 8.2. Używanie kontenerów inicjujących do ładowania kapsuł w zbiorach stanowych 210
  - 8.3. Żądanie pamięci masowej za pomocą szablonów PVC 216
  - 8.4. Uruchamianie zadań konserwacyjnych za pomocą kontrolerów Job i CronJob 222
  - 8.5. Wybór platformy dla aplikacji stanowych 229
  - 8.6. Laboratorium 231
9. *Zarządzanie wydawaniem nowych wersji aplikacji za pomocą rolloutów i rollbacków* 233
  - 9.1. Jak Kubernetes zarządza rolloutami? 234
  - 9.2. Aktualizowanie wdrożeń za pomocą rolloutów i rollbacków 237
  - 9.3. Konfigurowanie dla wdrożeń aktualizacji kroczących 245
  - 9.4. Aktualizacje kroczące w zbiorach demonów i zbiorach stanowych 254
  - 9.5. Strategie wydawania nowych wersji 259
  - 9.6. Laboratorium 261
10. *Pakowanie aplikacji i zarządzanie nimi za pomocą menedżera pakietów Helm* 263
  - 10.1. Jakie funkcjonalności Helm dodaje do Kubernetesa? 264
  - 10.2. Pakowanie własnych aplikacji za pomocą menedżera pakietów Helm 270
  - 10.3. Modelowanie zależności w wykresach 279
  - 10.4. Wykonywanie uaktualnień i rollbacków wydań Heima 283
  - 10.5. Zastosowania menedżera pakietów Helm 290
  - 10.6. Laboratorium 291
11. *Tworzenie aplikacji — programistyczne przepływy pracy oraz potok CI/CD* 292
  - 11.1. Programistyczny przepływ pracy oparty na Dockerze 293
  - 11.2. Programistyczny przepływ pracy Kubernetesa jako usługi 298
  - 11.3. Izolowanie obciążeń roboczych za pomocą kontekstów i przestrzeni nazw 305
  - 11.4. Ciągłe dostarczanie w Kubernetesie bez Dockera 310

- 11.5. Ocena programistycznych przepływów pracy w Kubernetesie 316
- 11.6. Laboratorium 318

### **TYDZIEŃ III. PRZYGOTOWANIE DO DZIAŁANIA W ŚRODOWISKU PRODUKCYJNYM 321**

- 12. *Konfigurowanie samonaprawiających się aplikacji 323*
  - 12.1. Routowanie ruchu do zdrowych kapsuł przy użyciu sond gotowości 324
  - 12.2. Wykorzystanie sond żywotności do restartowania kapsuł, które uległy awarii 330
  - 12.3. Bezpieczne wdrażanie uaktualnień za pomocą menedżera pakietów Heim 335
  - 12.4. Chronienie aplikacji i węzłów za pomocą limitów zasobów 342
  - 12.5. Ograniczenia samonaprawiających się aplikacji 350
  - 12.6. Laboratorium 351
- 13. *Centralizacja dzienników za pomocą oprogramowania Fluentd i Elasticsearch 352*
  - 13.1. Jak Kubernetes przechowuje wpisy dzienników? 353
  - 13.2. Gromadzenie dzienników z węzłów za pomocą Fluentd 357
  - 13-3- Wysyłanie dzienników do Elasticsearch 364
  - 13.4. Parsowanie i filtrowanie wpisów dzienników 369
  - !3-5- Opcje rejestrowania w Kubernetesie 374
  - 13.6. Laboratorium 376
- 14. *Monitorowanie aplikacji i Kubernetesa za pomocą pakietu narzędziowego Prometheus 377*
  - 14.1. Jak Prometheus monitoruje obciążenia robocze Kubernetesa? 378
  - 14.2. Monitorowanie aplikacji zbudowanych przy użyciu bibliotek klienckich Prometheusa 384
  - 14.3. Monitorowanie zewnętrznych aplikacji przy użyciu eksporterów wskaźników 391
  - 14.4. Monitorowanie kontenerów i obiektów Kubernetesa 397
  - 14.5. Inwestycje w monitorowanie 402
  - 14.6. Laboratorium 404

15-	<i>Zarządzanie ruchem przychodzącym za pomocą obiektu Ingress</i>	405
15.1.	W jaki sposób Kubernetes routuje ruch za pomocą obiektu Ingress?	406
15.2.	Routing ruchu HTTP za pomocą reguł obiektu Ingress	411
15-3-	Porównanie kontrolerów ruchu przychodzącego	418
15.4.	Używanie obiektu Ingress do zabezpieczania aplikacji za pomocą protokołu HTTPS	428
15.5.	Obiekt Ingress i kontrolery ruchu przychodzącego	433
15.6.	Laboratorium	434
16.	<i>Zabezpieczanie aplikacji za pomocą reguł, kontekstów i sterowania dostępem</i>	436
16.1.	Zabezpieczanie komunikacji za pomocą reguł sieciowych	437
16.2.	Ograniczanie możliwości kontenerów za pomocą kontekstów bezpieczeństwa	445
16.3.	Blokowanie i modyfikowanie obciążeń roboczych za pomocą zaczepów sieciowych	451
16.4.	Sterowanie dostępem za pomocą silnika Open Policy Agent	458
16.5.	Kwestie bezpieczeństwa w Kubernetesie	466
16.6.	Laboratorium	467

#### **TYDZIEŃ IV. CZYSTY KUBERNETES W PRAKTYCE . . . . . 469**

17-	<i>Zabezpieczanie zasobów za pomocą kontroli dostępu opartej na rolach</i>	471
17.1.	Jak Kubernetes zabezpiecza dostęp do zasobów?	472
17.2.	Zabezpieczanie dostępu do zasobów wewnątrz klastra	479
17.3.	Wiązanie ról z grupami użytkowników i kont usług	488
17.4.	Wykrywanie i kontrolowanie uprawnień za pomocą wtyczek	496
17.5.	Planowanie strategii RBAC	500
17.6.	Laboratorium	501
18.	<i>Wdrażanie Kubernetesa: klastry wielowęzłowe i wieloarchitekturowe</i>	503
18.1.	Co się znajduje w klastrze Kubernetesa?	504
18.2.	Inicjowanie płaszczyzny sterowania	508
18.3.	Dodawanie węzłów i uruchamianie obciążeń roboczych na węzłach linuksowych	512
18.4.	Dodawanie węzłów Windowsa i uruchamianie hybrydowych obciążeń roboczych	520

- 18.5. Kubernetes na dużą skalę 529
- 18.6. Laboratorium 530
- 19. *Kontrolowanie rozmieszczania obciążeń roboczych i automatyczne skalowanie 531*
  - 19.1. Jak Kubernetes rozdysponowuje obciążenia robocze? 532
  - 19.2. Zarządzanie rozmieszczeniem kapsuł za pomocą powinowactwa i antypowinowactwa 537
  - 19.3. Kontrolowanie wydajności za pomocą automatycznego skalowania 545
  - 19.4. Ochrona zasobów za pomocą wyłączeń i priorytetów 552
  - 19.5. Mechanizmy zarządzania obciążeniami roboczymi 559
  - 19.6. Laboratorium 561
- 20. *Rozszerzanie Kubernetesa o niestandardowe zasoby i operatory 562*
  - 20.1. Jak rozszerzać Kubernetes za pomocą niestandardowych zasobów? 563
  - 20.2. Wyzwalanie przepływów pracy za pomocą niestandardowych kontrolerów 567
  - 20.3. Zarządzanie zewnętrznymi komponentami przy użyciu operatorów 575
  - 20.4. Budowanie operatorów dla własnych aplikacji 584
  - 20.5. Kiedy rozszerzać Kubernetes? 591
  - 20.6. Laboratorium 592
- 21. *Uruchamianie w Kubernetesie funkcji bezserwerowych 594*
  - 21.1. Jak działają platformy bezserwerowe w Kubernetesie? 595
  - 21.2. Wywoływanie funkcji za pomocą żądań HTTP 601
  - 21.3. Wywoływanie funkcji za pomocą zdarzeń i harmonogramów 606
  - 21.4. Tworzenie warstwy abstrakcji dla funkcji bezserwerowych przy użyciu projektu Serverless 613
  - 21.5. Kiedy stosować funkcje bezserwerowe? 620
  - 21.6. Laboratorium 621
- 22. *Nauka nigdy się nie kończy 623*
  - 22.1. Dalsza lektura dla poszczególnych rozdziałów 623
  - 22.2. Wybór platformy Kubernetesa 627
  - 22.3. Jak jest zbudowany Kubernetes? 629
  - 22.4. Dołączanie do społeczności 630