SZKOLENIE ZAAWANSOWANE

Kubernetes w praktyce

KUBERNETES

Czas trwania: 5 dni (40h)

Cele szkolenia

- Wprowadzenie do wykorzystania platformy Kubernetes
- Poznanie dobrych praktyk związanych z wdrażaniem aplikacji wykorzystujących kontenery (w szczególności opartych o architekturę mikroserwisów)
- Nabycie umiejętności wykorzystania Kubernetes w zakresie monitorowania, tuningu i skalowania aplikacji

Zalety

- Szkolenie i prezentowane przykłady dostarczają ogromną dawkę praktycznych informacji
- Zdobywana wiedza ma dużą wartość merytoryczną i może być zastosowana podczas rozwiązywania problemów w rzeczywistych projektach
- Praktyka przed teorią - wszystkie szkolenia technologiczne prowadzone są w formie warsztatowej. Konieczna teoria jest wyjaśniana na przykładzie praktycznych zadań
- Konkretne umiejętności - w ramach każdego szkolenia rozwijamy praktyczne umiejętności związane z daną technologią i tematyką
- Nauka z praktykami - wszyscy trenerzy na co dzień pracują w projektach, gwarantuje to dostęp do eksperckiej wiedzy i praktycznego know-how

Dla kogo?

- Szkolenie dedykowane jest przede wszystkim deweloperom i administratorom, jednocześnie szeroko wprowadzające każdego zainteresowanego tworzeniem, uruchamianiem i zarządzaniem aplikacjami wykorzystującymi kontenery

Wymagania

- Praktyczna znajomość wybranej technologii kontenerowej np. Docker
- Wiedza z zakresu programowania, sieci komputerowych oraz systemu Linux

Program

1. Wprowadzenie do kontenerów na przykładzie platformy Docker
   a. Architektura, komponenty oraz wersjonowanie narzędzia Docker
   b. Budowa i optymalizacja obrazów aplikacji przy użyciu Dockerfile

   c. Rejestry obrazów - gdzie przechowywać paczki swoich aplikacji
   d. Ekosystem Docker - przegląd narzędzi powstałych wokół platformy Docker
   e. Codzienna administracja kontenerami - gromadzenie logów, metryki wydajności, polityki restartowania aplikacji
   f. Bezpieczeństwo kontenerów - co zrobić, aby zabezpieczyć system operacyjny oraz inne aplikacje przed złośliwym oprogramowaniem
   g. Omówienie różnic pomiędzy lokalną, a produkcyjną konfiguracją kontenerów
2. Architektura Kubernetes
   a. Komponenty klastra (masters oraz workers)
   b. Zarządzanie obiektami Kubernetes (imperatywne oraz deklaratywne)
   c. Manifesty obiektów - struktura oraz format (YAML)
   d. Docker w orkiestratorze Kubernetes
   e. Nowe wydania oraz okna wsparcia dla starszych wersji platformy
3. Podstawy konfiguracji
   a. Minikube jako najłatwiejszy sposób na instalację lokalnego klastra
   b. Dashboard czyli podgląd stanu klastra w przeglądarce internetowej
   c. Podział klastra na wirtualne przestrzenie (Namespaces)
   d. Kubernetes wykorzystujący maszyny fizyczne lub wirtualne? To nie ma znaczenia (Nodes)
   e. Pod jako podstawowa jednostka alokacji procesów w klastrze
   f. Labels and Selectors - grupowanie obiektów za pomocą etykiet
   g. Services - konfiguracja komunikacji pomiędzy aplikacjami w klastrze jak i dostęp do aplikacji ze świata zewnętrznego
   h. Zarządzanie liczbą replik aplikacji oraz sposobem ich aktualizacji za pomocą Deployments
   i. Jobs/CronJobs czyli sposób na uruchamianie zadań skryptowych w klastrze Kubernetes
   j. Konfiguracja centralnego monitoringu oraz logowania przy użyciu DaemonSets
   k. StatefulSets jako sposób na uruchamianie aplikacji stanowych w klastrze
4. Uwierzytelnianie oraz autoryzacja
   a. Organizacja informacji o klastrach i użytkownikach w pliku kubeconfig
   b. Przedstawienie typów użytkowników w Kubernetes: używanych przez administratorów (użytkownicy) oraz aplikacje (konta serwisowe)
   c. Omówienie strategii uwierzytelniania w klastrze, od użytkownika z hasłem, przez certyfikaty x509, po tokeny OpenID
   d. Kontrola dostępu oparta na rolach jako sposób na przypisywanie uprawnień do użytkowników
   e. Walidacja lub modyfikacja żądań za pomocą Admission Controllers
5. Sieci
   a. Porównanie architektury sieciowej: Docker vs. Kubernetes
   b. CNI jako interfejs służący do konfiguracji kart sieciowych kontenerów
   c. Omówienie różnych sposobów na publikacje aplikacji za pomocą Services (ClusterIP, NodePort, LoadBalancer, ExternalIP, ExternalName)
   d. Ingress czyli przekierowanie przychodzących zapytań HTTP do aplikacji uruchomionych w Kubernetes (na przykładzie kontrolera nginx)
   e. Blokada komunikacji sieciowej w klastrze za pomocą Network Policies
   f. Konfiguracja serwera rozwiązywania nazw w klastrze

6. Storage
    a. Zarządzanie zmiennymi środowiskowymi i plikami konfiguracyjnymi aplikacji z użyciem ConfigMaps
    b. Przechowywanie danych wrażliwych takich jak: hasła, klucze czy tokeny za pomocą Secrets (generic, docker-registry, tls)
    c. Utrwalanie danych z użyciem wolumenów różnego typu
    d. Dynamiczne oraz statyczne zarządzanie wolumenami w klastrze
7. Dodatkowe funkcjonalności
    a. Kontrolowanie poprawnego funkcjonowania aplikacji za pomocą różnego rodzaju próbówek (Liveness, Readiness oraz Startup probes)
    b. Konfiguracja automatycznego skalowania aplikacji ze względu na obciążenie przy użyciu Horizontal Pod Autoscalers
    c. Uruchamianie kontenerów/zadań przed startem głównej aplikacji
    d. Startowanie aplikacji na określonych maszynach oraz konfiguracja zależności pomiędzy uruchomionymi kontenerami (nodeSelector, affinity/antiAffinity, taints/tolerations)
    e. Zarządzanie zasobami klastra: minimalnymi oraz maksymalnymi limitami przypisanymi do kontenerów
    f. Priorytety aplikacji w klastrze oraz wywłaszczanie kontenerów z niskim priorytetem
    g. Utrzymanie maszyn klastra wraz z przygotowaniem okien obsługi
    h. Omówienie polityk aktualizacji aplikacji (recreate, ramped, blue/green, canary, a/b testing, shadow)
8. Logowanie oraz Monitoring
    a. Omówienie różnych architektur zbierania logów w klastrze Kubernetes wraz z przeglądem najpopularniejszych narzędzi
    b. Centralny monitoring zasobów w klastrze na przykładzie: Prometheus, AlertManager, Grafana
9. Bezpieczeństwo
    a. Testy penetracyjne klastra Kubernetes
    b. Ograniczenie uprawnień oraz kontrola dostępu aplikacji do komponentów systemu operacyjnego z użyciem SecurityContext
    c. Wymuszanie globalnych standardów specyfikacji aplikacji wykorzystując Pod Security Policy
10. Dystrybucje oraz użyteczne narzędzia
    a. Kubernetes jako usługa na przykładzie najpopularniejszych dostawców chmur publicznych (GKE, EKS, AKS)
    b. Zautomatyzowana instalacja klastra przy użyciu Kubernetes Operations (KOPS)
    c. Instalacja klastra Kubernetes "on premise" (Kubespray)
    d. Kubeless, Kubeapps, kubectx, kubens i inne narzędzia użyteczne w codziennej pracy z klastrem Kubernetes
11. Helm jako natywny menedżer pakietów aplikacji w Kubernetes