Open Policy Agent

Jakub Radek, Edyta Paruch, Andrzej Starzyk, Roksana Cieśla

2024

1 Introduction

Celem projektu jest przedstawienie możliwości technologii silnika Open Policy Agent - OPA w kontekście integracji z istniejącą aplikacją. W tym sprawozdaniu skupimy się na analizie procesu integracji OPA z istniejącą już aplikacją, obejmując projektowanie, implementację i ocenę sensowności oraz efektywności takiego rozwiązania. Poprzez praktyczne zastosowanie OPA w kontekście rzeczywistych aplikacji, będziemy badać potencjalne korzyści i wyzwania związane z jego wdrożeniem. W ramach projektu szczególną uwagę poświęcimy zrozumieniu mechanizmów działania OPA oraz jego możliwości konfiguracyjnych w kontekście konkretnych przypadków użycia. Analiza ta pozwoli nam na lepsze zrozumienie roli, jaką może odegrać OPA w zapewnianiu bezpieczeństwa oraz kontroli dostępu w środowiskach aplikacji.

2 Theoretical background/technology stack

2.1 Polityki

Polityka to zbiór zasad, zgodnie z którymi jest zarządzany pewien software'owy serwis. Mogą dotyczyć ruchu sieciowego, dostępu do serwerów, uprawnień użytkowników itp. Zazwyczaj są na sztywno zapisane w plikach danego serwisu.

2.2 OPA

Celem Open Policy Agent jest oddzielenie zarządzania politykami od serwisu. Oznacza to, że definicje polityk są umieszczone na osobnym serwerze, który udostępnia REST API. Za jego pomocą serwisy mogą wysyłać zapytania o to, czy podejmowane działania są zgodne z przyjętymi politykami.

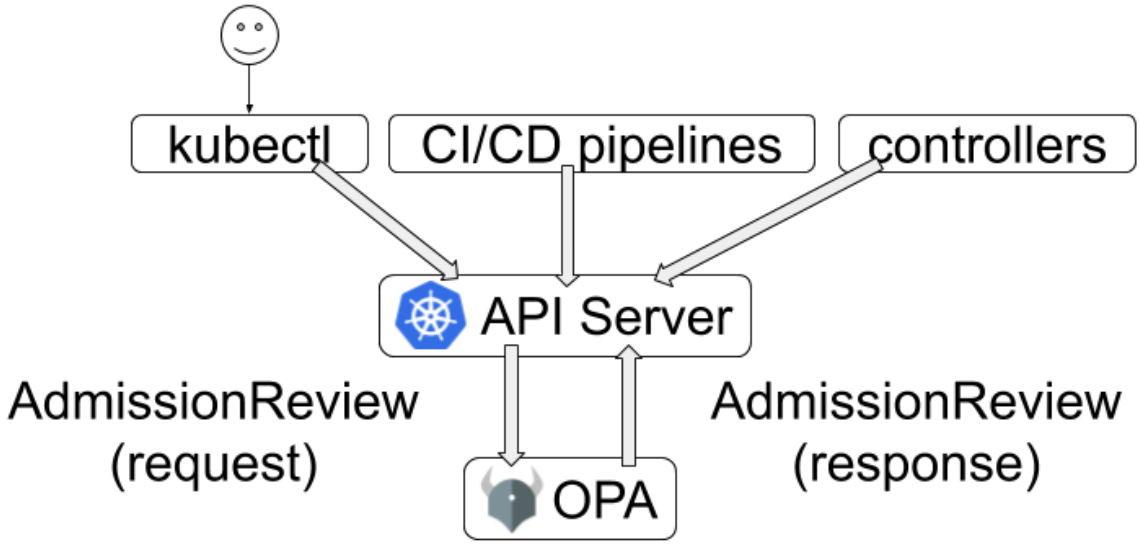
2.3 Definiowanie polityk

OPA przechowuje strukturę całego systemu w formacie JSON. Dzięki temu może łatwo sprawdzić, czy stan systemu jest zgodny z polityką. Te z kolei są definiowane za pomocą deklaratywnego języka REGO. Każda definicja składa się z pewnych wyrażeń logicznych. Jeśli wszystkie są prawdziwe, to założenia polityki są spełnione.

2.4 Integracja z Kubernetes

OPA można wdrożyć jako Admission Controller, który modyfikuje zapytania docierające do API Serwera Kubernetes. W ten sposób gdy jakikolwiek obiekt jest tworzony, modyfikowany lub usuwany API Serwer wysyła opis tej sytuacji jako żądanie do OPA. Ten serwis z kolei traktuje to dane w formacie JSON podobnie jak opisaną powyżej strukturę systemu - sprawdza zgodność z politykami i odsyła odpowiedź. API Serwer podejmuje na tej podstawie decyzję o przeprowadzeniu pewnej operacji lub w przeciwnym wypadku obsługuje odmowę. Wdrożenie tego rozwiązania polega na napisaniu kodu polityk, stworzeniu serwera OPA i wdrożeniu go do kubernetes.

Innym rozwiązaniem jest OPA Gatekeeper. Podobnie jak powyższe rozwiązanie pozwala na rozdzielenie polityk od serwera API i realizuje jako admission controller modyfikujący żądania. Jednak definiowanie polityk polega na configurowaniu zamiast kodowania, a rozstrzyganie zgodności z politykami bierze pod uwagę cały system, nie tylko pojedyńcze żądanie. Z uwagi na fakt, że jest to osobny projekt, choć powiązany z OPA, traktujemy to rozwiązanie jak ewentualny kierunek rozwoju case study.



Rysunek 1: Schemat integracji OPA i Kubernetes

3 Case study concept description

Integracja OPA z Kubernetesem jest jednym z głównych zastosowań tego narzędzia. Z tego powodu celem case study jest zaprezentowanie możliwości OPA w środowisku Kubernetes. Realizacja tego projektu zakłada zdefiniowanie polityk OPA dla przykładowej aplikacji w kontenerze na Kubernetesie. Polityki te pomogą zarządzać aplikacją i pokażą rozwiązania potencjalnych problemów wiążących się z udostępnianiem takiej aplikacji.

Do realizacji case study wykorzystana zostanie aplikacja biura turystycznego, powstała podczas prac nad projektem inżynierskim jednego z członków zespołu. Aplikacja realizuje funkcjonalności związane z proponowaniem wycieczek, posiada system logowania, zapisywania oraz wczytywania wycieczek. Nie jest ona jednak w pełni dostosowana do publicznego udostępnienia. Zaistniałymi problemami są między innymi potencjalnie zbyt duża liczba użytkowników korzystających z aplikacji jednocześnie, co może powodować niepożądane zachowania. Problem ten można zmniejszyć, tworząc kilka serwerów aplikacji i ustalając odpowiednie polityki dostępu, osobny dostęp dla pracowników biura, deweloperów aplikacji i użytkowników zewnętrznych oraz zapewnienie load balacingu pomiędzy tymi serwerami. Inny scenariusz wykorzystania polityk może dotyczyć tworzenia i usuwania kontenerów na potrzeby różnych konkretnych grup użytkowników oraz ich autoryzacji - przykładowo wymuszenie zalogowania, czy ograniczenie oferowanych funkcjonalności aplikacji dla innych grup użytkowników.

Integracja aplikacji z OPA zostanie wykonana przy użyciu standardowego kube-mgmt - komend i skryptów definiujących zasoby, polityki oraz ich wdrażanie w systemie.