Infrastructure as Code

mgr inż. Jakub Woźniak

Zarządzanie Systemami Rozproszonymi
Instytut Informatyki
Politechnika Poznańska

Wprowadzenie do Infrastructure as Code (IaC)

- Infrastructure as Code (IaC) to podejście, które pozwala zarządzać infrastrukturą za pomocą kodu, eliminując ręczne zarządzanie konfiguracją.
- IaC umożliwia zapisywanie, wersjonowanie i współdzielenie infrastruktury podobnie jak kodu aplikacyjnego.
- IaC pozwala na przywracanie infrastruktury do spójnego stanu oraz wdrażanie środowisk w sposób powtarzalny i audytowalny.
- Deklaratywne podejście IaC określa 'co' chcemy osiągnąć, a nie 'jak' – narzędzie wykonuje wszystkie potrzebne kroki.

Korzyści z IaC

1. Automatyzacja i eliminacja błędów ludzkich: Automatyzacja redukuje błędy wynikające z ręcznej konfiguracji.
2. Spójność środowisk: IaC gwarantuje identyczność środowisk – od deweloperskiego po produkcyjne.
3. Audytowalność i zgodność: Kod IaC jest wersjonowany, co ułatwia audytowanie zmian i zgodność z politykami bezpieczeństwa.
4. Szybkie przywracanie i odtwarzanie: Można przywrócić środowisko do ostatniej stabilnej wersji z repozytorium.
5. Łatwiejsze skalowanie: Automatyczne skalowanie zasobów dostosowuje infrastrukturę do zmieniających się potrzeb.

Co to jest Terraform?

- Terraform to narzędzie open-source do zarządzania infrastrukturą jako kod stworzone przez HashiCorp.
- Pozwala tworzyć i zarządzać infrastrukturą na różnych platformach chmurowych, w tym AWS, Azure, Google Cloud.
- Terraform definiuje infrastrukturę za pomocą deklaratywnego języka HCL (HashiCorp Configuration Language), opisując zasoby.
- Zarządzanie infrastrukturą jest oparte na stanie (`state`), co pozwala na porównanie pożądanego i aktualnego stanu zasobów.
- Terraform po zmianach licencyjnych został „sforkowany” do projektu OpenTofu.

Jak działa Terraform?

- Planowanie i stosowanie zmian: Proces obejmuje trzy kroki – pisanie kodu, planowanie (`terraform plan`) oraz wdrażanie (`terraform apply`).
- Krok 1: `terraform init` – inicjalizuje konfigurację i pobiera zależności.
- Krok 2: `terraform plan` – generuje plan pokazujący, jakie zasoby będą utworzone, zmienione lub usunięte.
- Krok 3: `terraform apply` – wdraża planowane zmiany, tworząc i konfigurując zasoby.
- Deklaratywne podejście zapewnia, że każdy zasób zostanie zaktualizowany zgodnie z oczekiwaniami.

Tworzenie infrastruktury na różnych platformach chmurowych

- Terraform wspiera zasoby chmurowe, takie jak EC2 w AWS, VM w Azure, Kubernetes, bazy danych, load balancery, VPC i więcej.
- Przykład: instancja EC2 na AWS zdefiniowana przez typ instancji, region i tagi.
- Jedna konfiguracja może być uruchamiana na różnych platformach dzięki dostawcom (providers) Terraform, np. `provider "aws"`, `provider "google"`.

Wprowadzenie do modułów w Terraform

- Moduły Terraform to wielokrotnego użytku zestawy plików konfiguracyjnych, umożliwiające tworzenie powtarzalnych elementów infrastruktury.
- Moduły organizują infrastrukturę strukturalnie, co ułatwia skalowanie i współpracę.
- Moduły mogą być tworzone samodzielnie lub pobierane z Terraform Registry – repozytorium gotowych modułów.

Tworzenie i używanie modułów

- Moduły składają się z plików konfiguracyjnych zawierających zmienne wejściowe (variables), zmienne wyjściowe (outputs) oraz zasoby (resources).
- Podstawowa struktura: folder modułu zawiera plik `main.tf` (zasoby), `variables.tf` (zmienne) i opcjonalny `outputs.tf`.
- Moduły są przydatne, gdy trzeba wielokrotnie tworzyć zasoby w różnych środowiskach, np. VPC dla dev, staging i prod.
- Moduły pozwalają na bardziej zwięzłą i czytelną konfigurację infrastruktury.

Stan infrastruktury (Terraform State)

- Terraform `state` to plik (zwykle `terraform.tfstate`) przechowujący aktualny stan infrastruktury zarządzanej przez Terraform.
- Plik stanu umożliwia śledzenie istniejących zasobów i stosowanie tylko niezbędnych zmian – aktualizując zasoby.
- Stan może być przechowywany lokalnie lub zdalnie (np. w Amazon S3), co umożliwia zespołom współpracę.
- Przykłady komend: `terraform show` (wyświetla stan), `terraform refresh` (aktualizuje stan).

Wprowadzenie do Terraform Cloud

- Terraform Cloud to zarządzana usługa HashiCorp, umożliwiająca zdalne przechowywanie stanu oraz współpracę zespołową.
- Zalety: zdalne przechowywanie stanu, automatyczne plany, współpraca zespołowa, audyt zmian.
- Terraform Cloud wspiera CI/CD i automatyczne wdrożenia z kontrolą dostępu do środowisk.
- Oferuje wersjonowanie i role dostępu, co zwiększa bezpieczeństwo zarządzania infrastrukturą w zespołach.

Terraform Cloud vs Atlantis (open-source)

- Atlantis to open-source narzędzie automatyzujące Terraform w repozytoriach Git (GitHub, GitLab).
- Pozwala na automatyczne generowanie `terraform plan` i `terraform apply` w odpowiedzi na pull requesty, integrując się z cyklem DevOps.
- Atlantis jest elastyczny, open-source, wymaga jednak konfiguracji w porównaniu do Terraform Cloud.
- Monitoruje zmiany w kodzie, wykonując `terraform plan` po pull requestach, widoczny do przeglądu przed zatwierdzeniem.

Przykład przepływu pracy z Atlantis

Typowy przepływ pracy w Atlantis obejmuje:

1. Zmiana kodu infrastruktury i pull request.
2. Atlantis generuje `terraform plan`, umożliwiając przegląd zmian przez zespół.
3. Po zatwierdzeniu Atlantis wykonuje `terraform apply`, wdrażając zmiany.

Podejście automatyzuje infrastrukturę przy użyciu GitOps, zwiększając bezpieczeństwo i przejrzystość.