Kurs DevOps

Lista 5

12 i 13 listopada 2025

Jako że zazwyczaj w dużych grupach znajduje się co najmniej jeden użytkownik Windowsa, to informuje się, iż zadania należy wykonywać z linii poleceń, o ile nie powiedziano inaczej w treści.

Używanie pluginów/rozszerzeń open-source do Ansible i Kubernetesa jest dozwolone, o ile w treści zadania nie powiedziano inaczej.

Zadanie 1.
Przy pomocy poleceń ansible *ad-hoc* pokaż, w jaki sposób zainstalować na zdalnej maszynie program dostarczający serwis systemd, a następnie w jaki sposób aktywować (ang. enable) ten serwis. Postaraj się nie używać modułów `shell` i `command`, a zamiast tego korzystać z modułów specyficznych dla zadań jakie chcesz wykonać (np. moduł `apt`).

W jaki sposób z użyciem ansible prosto sprawdzić najważniejsze informacje o maszynie, do której się łączymy (liczba rdzeni, podłączone dyski, wersja jądra itp.)?

Zadanie 2.
Wyjaśnij, jakie zastosowanie mają zmienne zdefiniowane w inwentarzu ansibla. Zaprezentuj, jak skonfigurować zmienne w tym samym pliku co są definiowane maszyny, a także jak to zrobić w osobnym pliku. Pokaż, w jaki sposób dodać jeden komputer do dwóch różnych grup.

Zadanie 3.
Uruchom kilka (co najmniej 2) niezależnych dockerów, a następnie zbuduj i zainstaluj przykładową aplikację napisaną w C/C++ na każdym z nich przy pomocy playbooka ansible. Pokaż, w jaki sposób wykorzystać zmienne dodawane do inwenatrza w celu określenia flag kompilacji specyficznych dla hosta.

Zadanie 4.
Przygotuj playbooka ansible, który będzie automatyzował następujące czynności związane z zarządzaniem stacjami roboczymy zawodników na Olimpiadzie Informatycznej. Zadbaj o to, by poszczególne zadania można było uruchamiać niezależnie od siebie.

- Zapakuj i skompresuj katalog z komputera zawodnika, a następnie pobierz go na węzeł zarządzający.
- Wyczyść katalog domowy na komputerze zawodnika i zainicjalizuj go przy pomocy archiwum.
- Wrzuć pliki z zadaniami do katalogu na komputerze zawodnika.
- Uruchom testową kompilację C/C++ na komputerze zawodnika.
- Pobierz logi journala oraz dmesg z komputera zawodnika na węzeł zarządzający.

Zadanie 5.
Zademonstruj, w jaki sposób pisać kod Ansible w sposób modularny, czyli między innymi:

- Jak przechowywać osobno zmienne?
- Jak podzielić playbook na mniejsze części (zawierające taski) nadające się do wielokrotnego użycia?

Zadanie 6.
Czym są pluginy Ansible? Zapoznaj się z listą pluginów, zapewniających nowe moduły, definicje inwentarza i filtry, dostarczanych w standardowej instalacji Ansible i pokaż najciekawsze.

Wypisz listę zainstalowanych na Twoim komputerze pluginów Ansible służących do łączenia się z maszynami. Jakie typy połączeń wspiera Ansible poza ssh?

Zadanie 7.
Zademonstruj, w jaki sposób przy użyciu Ansible Vault oraz pluginu `become` zmienić użytkownika na hoście przekazując hasło tego użytkownika w bezpieczny sposób.

Pokaż, jak zabezpieczyć hasło do Ansible Vault przy pomocy GPG (tak by gpg-agent zapewniał cachowanie).

Zadanie 8.
Wyjaśnij, w jaki sposób Ansible decyduje o tym jaką wartość będzie miała zmienna `zmienna_smok` w przykładach a) i b).
a)

```
mojagrupa:
  hosts:
    docker1:
      ansible_host: 172.17.0.2
      ansible_user: root
    docker2:
      ansible_host: 172.17.0.3
      ansible_user: root
grupa_b:
  hosts:
    docker1:
    docker2:
  vars:
    zmienna_smok: b
grupa_a:
  hosts:
    docker1:
    docker2:
  vars:
    zmienna_smok: a
```

b) Przy uruchomieniu `ansible-inventory -i inventory3.yaml -i inventory2.yaml --list`.
inventory2.yaml:

```
all:
  vars:
    zmienna_smok: 2
  children:
    mojagrupa:
      hosts:
        docker1:
          ansible_host: 172.17.0.2
          ansible_user: root
        docker2:
          ansible_host: 172.17.0.3
          ansible_user: root
```

inventory3.yaml:

```
all:
  vars:
    zmienna_smok: 3
  children:
```

```
mojagrupa:
  hosts:
    docker1:
      ansible_host: 172.17.0.2
      ansible_user: root
    docker2:
      ansible_host: 172.17.0.3
      ansible_user: root
```

c) Czy możesz uruchomić polecenie ad-hoc `head -n 100 /dev/random | tr -d -c a-z | grep -c ab` z użyciem modułu `command`? Co trzeba zrobić by pipelining zadziałał?
d) Czemu możemy chcieć w playbooku zastosować poniższą ścieżkę?

```
- hosts: monitoring
  tasks: []
```