

Spis treści

Przedmowa (11)

Wstęp do drugiego wydania (13)

Wstęp do pierwszego wydania (15)

1. Wprowadzenie (19)

- Uwaga do wersji (20)
- Do czego nadaje się Ansible? (20)
- Jak działa Ansible? (21)
- Na czym polega wielkość Ansible? (22)
- Czy Ansible nie jest zbyt proste? (26)
- Co muszę wiedzieć? (26)
- Czego tu nie znajdziesz? (27)
- Instalacja Ansible (27)
- Konfiguracja serwera testowego (28)
- Co dalej? (35)

2. Scenariusze: pierwsze kroki (37)

- Wstępne wymagania (37)
- Bardzo prosty scenariusz (38)
- Uruchomienie scenariusza (41)
- Scenariusz to plik YAML (42)
- Anatomia scenariusza (45)
- Czy coś się zmieniło? Śledzenie stanu serwera (49)
- Coś ciekawszego: szyfrowanie TLS (49)

3. Ewidencja: opisywanie serwerów (57)

- Plik ewidencyjny (57)
- Wstępne wymagania: kilka maszyn Vagrant (58)
- Funkcjonalne parametry ewidencji (60)
- Grupy, grupy i jeszcze raz grupy (62)
- Zmienne serwerowe i grupowe w pliku ewidencyjnym (66)
- Zmienne serwerowe i grupowe w osobnych plikach (67)
- Dynamiczna ewidencja (69)
- Podział ewidencji na kilka plików (73)
- Dodawanie wpisów w trakcie działania scenariusza za pomocą modułów `add_host` i `group_by` (74)

4. Zmienne i fakty (77)

- Definiowanie zmiennych w scenariuszu (77)
- Wyświetlanie wartości zmiennych (78)
- Rejestrowanie zmiennych (78)
- Fakty (81)

- Definiowanie nowej zmiennej za pomocą modułu `set_fact` (84)
- Wbudowane zmienne (85)
- Definiowanie zmiennych w wierszu poleceń (87)
- Priorytety (88)

5. Mezzanine: nasza testowa aplikacja (89)

- Dlaczego wdrażanie aplikacji produkcyjnych jest skomplikowane? (89)

6. Instalacja Mezzanine za pomocą Ansible (95)

- Wyświetlanie zadań scenariusza (95)
- Układ zainstalowanych plików (96)
- Zmienne jawne i poufne (96)
- Instalowanie wielu pakietów za pomocą pętli (`with_items`) (98)
- Instrukcja `become` w zadaniu (99)
- Aktualizacja rejestru `apt` (99)
- Sprawdzenie projektu za pomocą modułu `git` (100)
- Instalacja Mezzanine i innych pakietów w środowisku wirtualnym (102)
- Krótka dygresja: skomplikowane argumenty w zadaniach (104)
- Konfiguracja bazy danych (106)
- Tworzenie pliku `local_settings.py` na podstawie szablonu (107)
- Polecenia `django-manage` (110)
- Uruchamianie własnych skryptów Pythona w kontekście aplikacji (111)
- Utworzenie plików konfiguracyjnych usług (113)
- Aktywacja konfiguracji serwera Nginx (115)
- Instalacja certyfikatów TLS (115)
- Instalacja zadania Twitter w harmonogramie `cron` (116)
- Cały scenariusz (117)
- Uruchomienie scenariusza na maszynie wirtualnej Vagrant (120)
- Diagnostyka (120)
- Instalacja Mezzanine na wielu serwerach (121)

7. Skalowanie scenariuszy: role (123)

- Podstawowa struktura roli (123)
- Przykład: role `database` i `mezzanine` (124)
- Stosowanie ról w scenariuszach (124)
- Zadania wstępne i końcowe (125)
- Rola `database` instalująca bazę danych (126)
- Rola `mezzanine` instalująca aplikację Mezzanine (128)
- Tworzenie plików i katalogów ról za pomocą narzędzia `ansible-galaxy` (132)
- Role zależne (132)
- Repozytorium Ansible Galaxy (133)

8. Zaawansowane scenariusze (135)

- Obsługa błędnie działających poleceń: instrukcje `changed_when` i `failed_when` (135)
- Filtry (138)

- Wyszukiwarki (140)
- Zaawansowane pętle (146)
- Sterowanie pętlami (149)
- Dołączanie plików (151)
- Bloki (153)
- Obsługa błędów za pomocą bloków (154)
- Szyfrowanie poufnych danych (156)

9. Dostosowywanie serwerów, przebiegów i procedur (159)

- Wzorce specyfikowania serwerów (159)
- Określanie grupy serwerów (160)
- Wykonywanie zadania na komputerze sterującym (160)
- Wykonywanie zadania na innym komputerze niż serwer (161)
- Wykonywanie zadania na kolejnych serwerach (161)
- Wykonywanie zadania w grupie serwerów (162)
- Jednokrotne wykonanie zadania (163)
- Strategie przebiegów (163)
- Zaawansowane procedury (166)
- Jawne gromadzenie faktów (173)
- Odczytywanie adresu IP serwera (174)

10. Wtyczki zwrotne (175)

- Wtyczki stdout (175)
- Inne wtyczki (178)

11. Przyspieszanie Ansible (185)

- Zwielokrotnienie sesji SSH (opcja ControlPersist) (185)
- Potokowanie (188)
- Zapamiętywanie faktów (189)
- Równoległe połączenia (192)
- Równoległe wykonywanie zadań za pomocą instrukcji async (193)

12. Własne moduły (195)

- Przykład: sprawdzenie, czy zewnętrzny serwer jest dostępny (195)
- Użycie modułu script zamiast tworzenia własnego modułu (195)
- Skrypt can_reach jako moduł (196)
- Gdzie umieszczać własne moduły? (196)
- Jak Ansible uruchamia moduły? (196)
- Oczekiwane wyniki (198)
- Tworzenie modułów w języku Python (199)
- Dokumentowanie modułu (209)
- Diagnozowanie modułu (210)
- Implementowanie modułu jako skryptu Bash (211)
- Określanie alternatywnego położenia powłoki Bash (212)
- Przykładowe moduły (213)

13. Vagrant (215)

- Przydatne opcje konfiguracyjne środowiska Vagrant (215)
- Prowizjoner Ansible (217)
- Kiedy jest uruchamiany prowizjoner? (217)
- Plik ewidencyjny tworzony przez środowisko Vagrant (217)
- Równoległe prowizjonowanie maszyn (218)
- Definiowanie grup maszyn wirtualnych (219)
- Lokalny prowizjoner Ansible (220)

14. Amazon EC2 (221)

- Terminologia (222)
- Poświadczenia (224)
- Warunek: instancja biblioteki Python Boto (225)
- Dynamiczna ewidencja (225)
- Definiowanie dynamicznych grup instancji na podstawie tagów (228)
- Chmury EC2-VPC i EC2-Classic (230)
- Przygotowanie pliku ansible.cfg do korzystania z chmury EC2 (231)
- Uruchamianie nowych instancji (231)
- Pary kluczy EC2 (233)
- Grupy bezpieczeństwa (234)
- Uzyskiwanie najnowszego obrazu AMI (236)
- Dodanie nowej instancji do grupy (237)
- Oczekiwanie na uruchomienie instancji (239)
- Idempotentne tworzenie instancji (240)
- Wszystko razem (241)
- Definiowanie chmury VPC (242)
- Tworzenie obrazów AMI (246)
- Inne moduły (250)

15. Docker (253)

- Przykład użycia programów Docker i Ansible (254)
- Czas życia aplikacji Docker (254)
- Przykładowa aplikacja: Ghost (255)
- Nawiązywanie połączenia z demonem Dockera (255)
- Uruchomienie kontenera na lokalnym komputerze (256)
- Tworzenie obrazu za pomocą pliku Dockerfile (257)
- Orkiestracja kilku kontenerów na lokalnym komputerze (258)
- Wysyłanie obrazów do rejestru Dockera (259)
- Odczytywanie informacji o lokalnych obrazach (260)
- Instalowanie aplikacji zawartych w obrazach (262)
- Ansible Container (266)

16. Diagnostowanie scenariuszy (273)

- Czytelne komunikaty o błędach (273)
- Diagnostowanie połączenia SSH (274)
- Moduł debug (275)

- Debugger scenariuszy (275)
- Moduł assert (277)
- Sprawdzenie scenariusza przed uruchomieniem (279)
- Wybieranie zadań do wykonania (280)

17. Zarządzanie serwerami Windows (283)

- Połączenie z systemem Windows (283)
- PowerShell (284)
- Moduły Windows (286)
- Nasz pierwszy scenariusz (286)
- Aktualizacja systemu Windows (287)
- Tworzenie lokalnych kont użytkowników (288)
- Podsumowanie (291)

18. Ansible i urządzenia sieciowe (293)

- Obecny stan modułów sieciowych (293)
- Lista obsługiwanych urządzeń (294)
- Przygotowanie urządzenia sieciowego (294)
- Jak funkcjonują moduły? (297)
- Pierwszy scenariusz (297)
- Ewidencja i zmienne w modułach sieciowych (298)
- Korzystanie z plików konfiguracyjnych (302)
- Szablony, szablony, szablony (305)
- Gromadzenie faktów (306)
- Podsumowanie (307)

19. Ansible Tower: wersja dla firm (309)

- Modele subskrypcji (309)
- Jakie problemy rozwiązuje Ansible Tower? (311)
- Interfejs REST API (316)
- Interfejs CLI (316)
- Co dalej? (320)

A. Protokół SSH (321)

- Natywny klient SSH (321)
- Agent SSH (321)
- Uruchomienie agenta SSH (322)
- Przekazywanie agenta (323)
- Klucze hosta (326)

B. Role IAM i poświadczenia EC2 (329)

- Konsola AWS Management Console (329)
- Wiersz poleceń (330)

Słowniczek (333)

Bibliografia (339)

Skorowidz (341)