УДК: 004.734:004.053:004.75

АВТОМАТИЗАЦІЯ РОЗГОРТАННЯ ТА НАЛАШТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНФРАСТРУКТУРИ СТВОРЕНОЇ В СЕРЕДОВИЩІ ХМАРНИХ ОБЧИСЛЕНЬ

студент, Обозний Д. М.

https://orcid.org/0000-0003-0108-4587

e-mail: dobozniy@gmail.com

студентка, Поштацька К. В.

https://orcid.org/0000-0002-0902-3722

e-mail: poshtatskayakatirina@gmail.com

Національний технічний університет України «Київський політехнічний інститут ім. Ігоря Сікорського», Україна, Київ

В даній роботі запропоновано метод автоматизованого розгортання інфраструктури, впровадження та конфігурація програмного забезпечення в середовищах хмарних обчислень. Метод дозволяє описувати інфраструктурні об'єкти за допомогою коду та динамічно модифікувати конфігурації в залежності від хмари. Запропонований метод використовує утиліти Ansible та Terraform для взаємодії з API середовища та віртуальними машинами. Розроблений метод дозволяє паралельно налаштовувати віртуальні машини та мати ідентичну інфраструктуру в різних середовищах хмарних обчислень. Тестування проводилося в середовищах AWS та Azure. Метод дозволяє пришвидшити розгортання інфраструктури та забезпечує ідемпотентність конфігурації програмного забезпечення.

Ключові слова: автоматизація, конфігурація, програмне забезпечення, середовище хмарних обчислень, розгортання інфраструктури.

студент, Обозный Д. Н., студентка Поштацкая Е. В. Автоматизация развертывания и настройки программного обеспечения инфраструктуры, созданной в среде облачных вычислений / Национальный технический университет Украины «Киевский политехнический институт имени Игоря Сикорского», Украина, Киев.

В данной работе предложен метод автоматизированного развертывания инфраструктуры, внедрение и конфигурация программного обеспечения в средах облачных вычислений. Метод позволяет описывать инфраструктурные объекты с помощью кода и динамично модифицировать конфигурации в зависимости от облака. Предложенный метод использует утилиты Ansible и Terraform для взаимодействия с API среды и виртуальными машинами. Разработанный метод позволяет параллельно настраивать виртуальные машины и иметь идентичную инфраструктуру в различных средах облачных вычислений. Тестирование проводилось в среде AWS и Azure. Метод позволяет ускорить развертывание инфраструктуры и обеспечивает идемпотентность конфигурации программного обеспечения.

Ключевые слова: автоматизация, конфигурация, программное обеспечение, среда облачных вычислений, развертывание инфраструктуры.

D. Oboznyi, student, K. Poshtatska, student, Automation of deployment and configuring software of the infrastructure created in the cloud environment / National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine, Kyiv.

This paper proposes a method of automated infrastructure deployment, implementation and configuration of software in cloud computing environments. The method allows you to describe

infrastructure objects using code and dynamically modify configurations depending on the cloud. This method uses the Ansible and Terraform utilities to interact with the cloud environment API and virtual machines. The developed method allows you to configure virtual machines in parallel and have an identical infrastructure in different cloud computing environments. Testing was performed in AWS and Azure environments. The method accelerates the deployment of the infrastructure and provides idempotence of the software configuration.

Key words: automation, configuration, software, cloud computing environment, infrastructure deployment.

Вступ. Розвиток інформаційних технологій призводить до збільшення кількості сфер діяльності людини, які перейшли в цифровий простір. Стрімке зростання та розширення інфраструктури потребує її швидкого налаштування з подальшою підтримкою оновлень. Велика кількість підприємств – представників малого та середнього бізнесу є стартапами з обмеженою кількістю фінансів, які можуть бути виділені на побудову власної інфраструктури. Постає питання пошуку оптимального рішення для швидкого, автоматичного налаштування інфраструктури, що не вимагатиме значних вкладень та буде доступною для підприємств, що розвиваються. Така можливість існує лише в середовищах хмарних обчислень. За допомогою них стає можливим гнучке конфігурування інфраструктури при будь-якому масштабі та потребах бізнесу.

Сьогодні широко використовуються утиліти для створення об’єктів в середовищах хмарних обчислень, проте більшість компаній не мають чіткого плану для швидкої зміни регіону збереження даних. До того ж в певних країнах та регіонах впроваджені закони, що накладають обмеження на збереження даних своїх резидентів поза

межами території власної країни. Оскільки з ростом кількості клієнтів зростає і масштаб інфраструктури, існує потреба автоматизації та підтримки великої кількості взаємозв'язаних між собою об'єктів.

Іншою проблемою, яка постає при проектуванні інфраструктури є гарантія використання безпечного програмного забезпечення. В наш час існує безліч видів шкідливого програмного забезпечення, дія якого може призвести до викрадення конференційних даних, руйнування системи в цілому. Кожен розроблений програмний продукт може містити вразливості у будь-якому вигляді, що можуть призвести до проникнення шкідливого програмного забезпечення. Такі вразливості відслідковуються та виправляються розробниками цього продукту у вигляді оновлень. Процес оновлень є доволі складною задачею тому що потребує динамічної зміни конфігурації в залежності від версій.

В даній роботі запропоновано метод комбінованого описання інфраструктури та конфігурацій програмного забезпечення для автоматизації процесу створення інфраструктури та встановлення програмного забезпечення. Також цей метод дозволяє зробити ідемпотентним процес оновлення інфраструктури та програмного забезпечення.

Постановка проблеми. Питання неконтрольованої зміни інфраструктури є актуальним в середовищах хмарних обчислень. У роботі [1] розглянуті описані проблеми, що доповнюються випадками використанням неактуальної документації, яка може бути неконсистентна до існуючого процесу створення об'єктів.

Кількість навантаження на інфраструктуру може змінюватись з плином часу, тому основним з питань є автоматизована зміна розміру інфраструктури в залежності від навантаження. У роботі [2] розглянутий метод балансування навантаження між

обчислювальними вузлами, проте не розглядається проблема відмови регіону певного середовища хмарних обчислень.

Для взаємодії з API середовищ хмарних обчислень використовуються такі утиліти, як AWS CloudFormation або Azure Resource Manager. Дані утиліти можуть використовуватись лише з хмарою, для якої вони були розроблені. Вони мають обмежену сумісність з іншим програмним забезпеченням чи сервісами, які знаходяться поза хмарою. Як зазначено у роботах [3, 4, 5], для налаштування програмного забезпечення на віртуальних машинах використовуються такі утиліти, як Chef та Puppet. Вони мають клієнт-серверну архітектуру та потребують більшої кількості, аніж сценарії Ansible, які потребують лише SSH з'єднання.

Щоб вирішити проблему автоматизації розгортки інфраструктури та налаштування програмного забезпечення, описану у роботі [6], розроблений метод використання маніфестів Terraform для опису інфраструктури в хмарі та використання Ansible плейбуків (сценаріїв) для розгортки та налаштування програмного забезпечення всередині середовища хмарних обчислень.

Опис розробленого алгоритму. Діаграма класів, що описує метод автоматизованої розгортки та налаштування інфраструктури зображена на рисунку 1.

Запропонований метод можна поділити на 2 частини: взаємодія з об'єктами середовищ хмарних обчислень (Terraform) та розгортка і налаштування програмного забезпечення в середовищі (Ansible).

Вхідними даними для розроблених модулів є характеристики мережі, характеристики віртуальних машин, мережевих сховищ даних, баз даних, прав доступу, систем моніторингу та логінгу, тип хмари.

На основі описаних даних, Terraform створює інфраструктуру в одному або декількох середовищах хмарних обчислень, визначаючи залежності між об'єктами. Створення об'єктів відбувається паралельно, враховуючи залежності. Таким чином, скорочується час розгортки інфраструктури до мінімального часу максимально довгої послідовності в графі залежності об'єктів. Існує можливість взаємодії об'єктів різних хмарних провайдерів між собою.

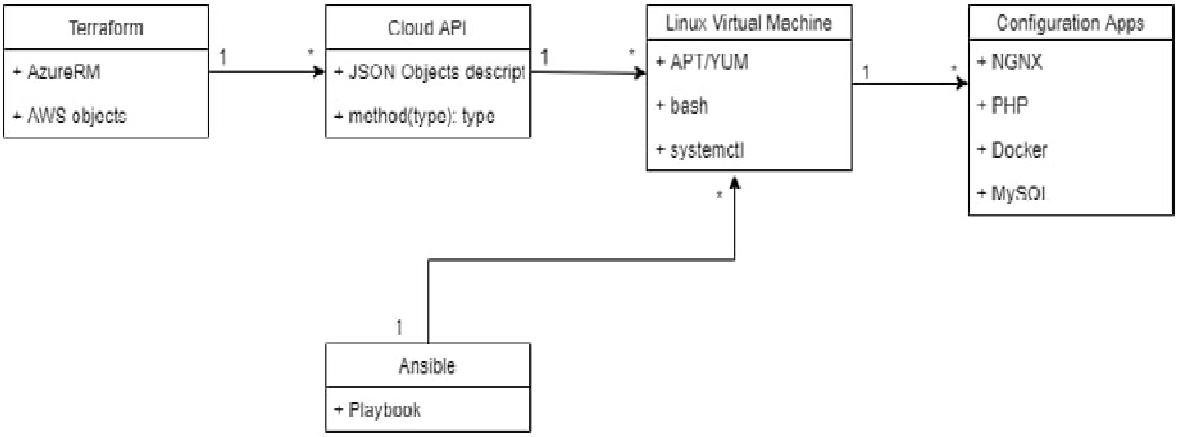

Рис. 1. Діаграма класів методу автоматизованої побудови інфраструктури

Після розгортки інфраструктури починається етап встановлення програмного забезпечення на обчислювальні вузли. В цьому випадку використовуються ролі Ansible, які являють собою модулі. Комбінація таких модулів використовується у сценаріях. Змінні, потрібні для кожної ролі, описуються в файлах змінних, які можуть бути груповими – для групи віртуальних машин або сервісів, а також індивідуальними – специфічні змінні характерні лише для окремого хосту. Для роботи Ansible потрібно мати лише приватну частину SSH ключа, публічна частина якого знаходиться на віртуальних машинах в хмарі.

В процесі виконання сценарію, Ansible за допомогою власних методів перевірки або методів, описаних в ролях, звіряє стан описаної дії з актуальною на віртуальній машині. Наприклад, при зміні прав доступу до файлу, Ansible порівнює актуальне значення прав,

тип файлу, власника та групу власників. У випадку відмінностей, він змінює файл до того стану, який описаний у ролі. Таке рішення є ідемпотентним – незалежно від кількості запусків сценарію результат буде одним та передбачуваним.

Послідовне виконання Terraform та Ansible реалізується у системі неперервної інтеграції та неперервної доставки GitLab CI. Даний підхід дозволяє відслідковувати зміни в коді та виконувати, описані скриптовою мовою сценарії запуску Terraform та Ansible, спрацьовуючи в залежності від змін у файлах репозиторію з кодом. Таким чином, процес зміни конфігурації, оновлення та доставки програмного забезпечення виконується автоматизовано. Розробник, або системний адміністратор не мають прав доступу для зміни об’єктів в хмарі або на віртуальних машинах всередині хмари. Такий підхід реалізує модель безпеки з нульовою довірою, як описано у роботі [7].

Тестування. Для тестування були створені маніфести Terraform для створення мережі, віртуальних машин, балансувачів навантаження та віртуальних машин. В якості програмного забезпечення, що розгорталось був використаний LEMP стек.

Terraform маніфести створювали ідентичну інфраструктуру в середовищах хмарних обчислень AWS та Azure в регіонах US West та EU Central. Для маршрутизації трафіку в різні регіони в залежності від географічного розташування клієнта був використаний сервіс AWS Route53 з налаштуванням Geo DNS. При відмові сервісів розгорнутих в одному з регіонів, трафік направлявся в інший регіон. Час розгортки інфраструктури в залежності від хмари, переналаштування та розгортки програмного забезпечення можна побачити в таблиці 1.

Таблиця

Часові параметри

Хмара	AWS (US West)	Azure (EU Central)
Час розгортки інфраструктрури, с	28.31	38.24
Час переналаштування регіонів, с	5	
Час розгортки програмного забезпечення, с	78,3	

Висновки. Розроблений метод дозволяє пришвидшити та оптимізувати процес доставки програмного продукту в різні географічні регіони, отримати контрольований процес оновлень, розгортки, підтримки інфраструктури та програмних продуктів всередині. Основною причиною розробки алгоритму є ускладнений процес підтримки та оновлення інфраструктури програмного забезпечення існуючими утилітами , а також неможливе швидке переключення або одночасне використання декількох середовищ хмарних обчислень. Тестування реалізованого методу показало оптимальні результати роботи і задовольнло основну причину його розробки.

Література:

1. Lindkvist C. (2013). Configuration Management in Complex Engineering Projects. Procedia CIRP, 2, 173-176.

2. Лабжинський В. А. (2018). Базові підходи та особливості використання хмарних, туманних і росистих обчислень. Вимірювальна та Обчислювальна Техніка в Технологічних Процесах, 4, 60-91.

3. Jourdan S, Pomes P. (2017). Infrastructure as Code (IAC) Cookbook. Packt Publishing.

4. Rahman A. (2019). A systematic mapping study of infrastructure as code research. Information and Software Technology, 1, 65-77.

5. Brouse, P.S. (2008). Configuration management in: A. Sage (Ed.), Systems engineering and management for sustainable development, 1, 214-242.

6. Brikman Y. (2019). Terraform: Up & Running.Boston: O'Reilly Media, Inc.

7. A paradigm shift for application security [Електронний ресурс] // Portshift. – 2019. – Режим доступу до ресурсу: https://www.portshift.io/solutions/zero-trust-security/.

References:

1. Lindkvist C. (2013). Configuration Management in Complex Engineering Projects. Procedia CIRP, 2, 173-176.

2. Labzhynskyi V. A. (2018). Bazovi pidkhody ta osoblyvosti vykorystannia khmarnykh, tumannykh i rosystykh obchyslen. Vymiriuvalna ta Obchysliuvalna Tekhnika v Tekhnolohichnykh Protsesakh, 4, 60-91. [in Ukrainian].

3. Jourdan S, Pomes P. (2017). Infrastructure as Code (IAC) Cookbook. Packt Publishing.

4. Rahman A. (2019). A systematic mapping study of infrastructure as code research. Information and Software Technology, 1, 65-77.

5. Brouse, P.S. (2008). Configuration management in: A. Sage (Ed.), Systems engineering and management for sustainable development, 1, 214-242.

6. Brikman Y. (2019). Terraform: Up & Running.Boston: O'Reilly Media, Inc.

7. A paradigm shift for application security [Elektronnyi resurs] // Portshift. – 2019. – Rezhym dostupu do resursu: https://www.portshift.io/solutions/zero-trust-security/.