

УДК 004.4'22

Миροнова Н.А.¹, Ліпихін С.О.², Терлецький С.В.², Денисенко С.В.²

¹ канд. техн. наук, доц. НУ «Запорізька політехніка»

² студ. гр. КНТ-138 НУ «Запорізька політехніка»

СУЧАСНІ ТЕХНОЛОГІЇ КОНТЕЙНЕРИЗАЦІЇ

Контейнеризація – це віртуалізація і ізоляція ресурсів на рівні операційної системи, яка дозволяє запускати застосунок і необхідний йому мінімум системних бібліотек в повністю стандартизованому контейнері, що з'єднують з хостом або чим-небудь зовнішнім по відношенню до нього за допомогою певних інтерфейсів.

Всі компоненти, необхідні для запуску програми, упаковуються як один образ і можуть бути використані повторно. Застосунок в контейнері працює в ізольованому середовищі і не використовує пам'ять, процесор або накопичувач пам'яті хостової операційної системи. Це гарантує ізолюваність процесів всередині контейнера.

Існують два основні варіанти віртуалізації, а точніше два підходи до створення незалежних ізольованих обчислювальних просторів на одному фізичному сервері: віртуальні машини і віртуальні контейнери. У першому випадку для кожної віртуальної машини використовується власна гостьова операційна система(ОС) і гіпервізор, а в другому – для всіх контейнерів застосовується ядро однієї хостової ОС.

Однак, оскільки віртуальні машини включають операційну систему, їх розмір може складати декілька гігабайт. Також недоліком віртуальних машин можна назвати те, що для завантаження ОС і ініціалізації програми, які в них розміщені, потрібно набагато більше часу. Контейнери займають менше простору і, в основному, їх розмір вимірюється в мегабайтах. Порівнюючи їх продуктивність з віртуальними машинами у часі, контейнери можуть запускатися майже миттєво.

Отже, основними перевагами контейнерів є:

– гнучке середовище:найбільша перевага у використанні технологій контейнерів полягає в тому, що їх можна створювати набагато швидше, ніж екземпляри віртуальних машин;

– підвищена продуктивність: контейнери збільшують ефективність розробників, за рахунок усунення міжмережевих залежностей і конфліктів. Кожен контейнер може розглядатися як окремий мікросервіс.

– управління версіями дозволяє відстежувати версії контейнера, стежити за відмінностями між ними та ін.;

– портативність середовища обчислень: контейнери інкапсулюють всі відповідні деталі, такі як залежності застосунків і операційні системи, необхідні для запуску програми. Це спрощує транспортування та тиражування контейнерів з одного середовища в інше;

– безпека: контейнери ізолюють процеси одного контейнера від іншого, та від базової інфраструктури. Таким чином, будь-яке оновлення або зміна в одному контейнері не впливає на інший контейнер.

Тим не менш, контейнери мають низку недоліків, таких як:

– підтримка NativeLinux: більшість контейнерних технологій, таких як Docker, засновані на Linux-контейнерах (LXC). Тому запуск цих контейнерів в середовищі Microsoft викликає небажані труднощі;

– контейнери – відносно нова технологія на ринку. Через нестачу кваліфікованих спеціалістів, на вирішення проблем витрачається більше часу і ресурсів;

– підвищена складність: при збільшенні числа контейнерів, які працюють з додатком, також збільшується коефіцієнт складності. Управління мережею контейнерів є складним та об'ємним процесом. Такі інструменти, як Kubernetes можуть полегшити управління великою кількістю контейнерів.

Отже, проведене дослідження націлене на актуалізацію та формалізацію основних відомостей про платформу контейнеризації Docker, виділення основних схем та підходів роботи з цим застосунком, а також перспективи та подальший розвиток даної технології у системі Kubernetes, як однієї з найбільш популярних та використовуваних на сьогоднішній день.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Hello Minikube [Electronic resource]. – Access mode: <https://kubernetes.io/docs/tutorials/hello-minikube/>.

2. Docker Overview [Electronic resource]. – Access mode: <https://docs.docker.com/engine/docker-overview/>.

3. Docker vs VMWare: How Do They Stack Up? [Electronic resource]. – Access mode: <https://www.upguard.com/articles/docker-vs.-vmware-how-do-they-stack-up>.