УДК 004.75

І. А. Гунько

ПЕРЕВАГИ ВИКОРИСТАННЯ ДОКЕРНИХ КОНТЕЙНЕРІВ ПЕРЕД ВІРТУАЛЬНИМИ МАШИНАМИ ДЛЯ ВІРТУАЛІЗАЦІЇ

Вінницький національний технічний університет

Анотація

Розглянуто сутність побудови додатків на основі контейнерів. Визначено переваги контейнеризації над використанням віртуальних машин.

Ключові слова: віртуалізація, контейнеризація, докер, контейнер, віртуальна машина, додаток, розгортання.

Abstract

The essence of building applications based on containers is considered. The advantages of containerization over the use of virtual machines are determined.

Keywords: virtualization, containerization, docker, container, virtual machine, application, deployment.

Вступ

Подібно віртуальній машині докер запускає свої процеси у власній, заздалегідь налаштованої операційній системі. Але при цьому всі процеси докера працюють на фізичному host сервері ділячи все процесори і всю доступну пам'ять з усіма іншими процесами, запущеними в host системі. Підхід, який використовується докером знаходиться посередині між запуском всього на фізичному сервері і повної віртуалізації, пропонованої віртуальними машинами. Цей підхід називається контейнеризацією.

Результати дослідження

Докер - це відкрита платформа для розробки, доставки і експлуатації додатків. Docker розроблений для більш швидкого викладання додатків. За допомогою docker можна відокремити додаток від інфраструктури і працювати з інфраструктурою як керованим додатком. У своєму ядрі docker дозволяє запускати практично будь-який додаток, безпечно ізольований в контейнері. Безпечна ізоляція дозволяє запускати на одному хості багато контейнерів одночасно. Легка природа контейнера, який запускається без додаткового навантаження гіпервізора, дозволяє досягати більшого від апаратної частини. Docker легкий і швидкий. Він надає стійку, рентабельну альтернативу віртуальним машинам на основі гіпервізора. Він особливо корисний в умовах високих навантажень, наприклад, при створення власного хмари або платформи-як-сервісу (platform-as-service). Він так само корисний для маленьких і середніх додатків, коли потрібно отримувати більше з наявних ресурсів.[1]

Контейнер складається з операційної системи, користувацьких файлів і метаданих. Кожен контейнер створюється з образу. Цей образ говорить docker-у, що знаходиться в контейнері, який процес запустити, коли запускається контейнер і інші конфігураційні дані. Docker образ доступний тільки для читання. Коли docker запускає контейнер, він створює рівень для читання / запису зверху образу (використовуючи union file system), в якому може бути запущено додаток. [2]

Використання Docker дозволяє швидше і ефективніше доставляти або переміщати код, стандартизує виконувані додатками операції і в цілому економить кошти, оптимізуючи використання ресурсів. Завдяки Docker користувачі отримують об'єкт, який з високою надійністю можна запускати на будь-якій платформі. Простий синтаксис Docker надає користувачам повний контроль над виконуваними операціями. Повсюдне впровадження контейнерів має на увазі доступ до різноманітних інструментів і готовим додатків, які можна використовувати з Docker.[1]

Контейнери Docker можна використовувати в якості основних компонентів для створення сучасних платформ і додатків. Docker спрощує збірку і запуск розподілених мікросервісних архітектур,

розгортання коду за допомогою стандартизованих конвеєрів безперервної інтеграції і доставки, створення високомасштабуємих систем обробки даних і повністю керованих платформ для розробників.[1]

Програмне забезпечення функціонує в середовищі Linux з ядром, що підтримує cgroups і ізоляцію просторів імен (namespaces); існують збірки тільки для платформ x86-64 і ARM. Починаючи з версії 1.6 можливе використання в ОС Windows. Для економії дискового простору проект використовує файлову систему Aufs з підтримкою технології каскадно-об'єднаного монтування: контейнери використовують образ базової операційної системи, а зміни записуються в окрему область. Також підтримується розміщення контейнерів в файлової системі Btrfs з включеним режимом копіювання при записі. До складу програмних засобів входить демон - сервер контейнерів, клієнтські засоби, що дозволяють з інтерфейсу командного рядка керувати образами і контейнерами, а також API, що дозволяє в стилі REST керувати контейнерами програмно. Демон забезпечує повну ізоляцію контейнерів на рівні файлової системи (у кожного контейнера власна коренева файлова система), на рівні процесів (процеси мають доступ тільки до власної файлової системи контейнера, а ресурси розділені засобами libcontainer), на рівні мережі (кожен контейнер має доступ тільки до прив'язаному до нього мережевого простору імен і відповідним віртуальним мережевим інтерфейсам).[2]

Висновки

У порівнянні з віртуальними машинами контейнери вимагають набагато менше ресурсів (наприклад, їм не потрібна повна ОС), їх простіше розгортати і вони швидше запускаються. Це дозволяє підвищити щільність розгортання, тобто запустити на одній одиниці обладнання більше служб і скоротити витрати на них. Запуск на одному ядрі призводить до того, що рівень ізоляції буде нижче, ніж на віртуальних машинах. Основна мета образу - привести середу (залежності) до однаковості в різних розгортаннях. Це означає, що можливо налагодити образ на одному комп'ютері, а потім розгорнути його на іншому комп'ютері і отримати те ж середовище. Образ контейнера - це спосіб упаковки додатку або служби для надійного і відтворюваного розгортання. Можна сказати, що Docker є не тільки технологією, але ще філософією і процесом.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Немет, Э. Руководство администратора Linux [Текст] = Linux Administration Handbook / Э. Немет, Г. Снайдер ; пер. с англ. – 5-е изд. – М. : ООО "И.Д. Вильямс", 2016. – 1072 с. – ISBN 978-5-8459-1093..

2. Tilkov, Stefan (17 November 2014). "How small should your microservice be?". innoq.com. Retrieved 4 January 2017.

3. Азарова А.О. Методичні вказівки до проведення практичних занять та до виконання самостійної індивідуальної роботи з дисципліни «Основи науково-дослідної роботи» для студентів напрямів підготовки 6.030601 – «Менеджмент» та 6.170103 – «Управління інформаційною безпекою» / Азарова А.О., Карпінець В.В. – Вінниця: ВНТУ, 2013. – 44 с.

Ілля Андрійович Гунько – студент групи УБ-18м, факультет менеджменту та інформаційної безпеки, Вінницький національний технічний університет, м. Вінниця, e-mail: illia.gunko69@gmail.com

Науковий керівник: Людмила Миколаївна Ткачук – канд. економічних наук, доцент кафедри МБІС, Вінницький національний технічний університет, м. Вінниця, e-mail: ludatkachuk2017@gmail.com

Illia Gunko - student of UB-18m group, faculty of Management and Information Security, Vinnitsa National Technical University, Vinnitsa, e-mail: illia.gunko69@gmail.com

Supervisor: Liudmyla Tkachuk – cand. economic sciences, Assistant Professor of Building MBIS, Vinnitsa National Technical University, Vinnitsa, e-mail: ludatkachuk2017@gmail.com