УДК 001.201

Фаррахов И.Г.
студент магистратуры
Набережночелнинский институт (филиал) КФУ
Россия, г. Набережные Челны
Якупов И.М.
студент магистратуры
Набережночелнинский институт (филиал) КФУ
Россия, г. Набережные Челны

АВТОМАТИЗИРОВАННЫЙ ИНСТРУМЕНТАРИЙ РАЗВЕРТЫВАНИЯ ОБЛАЧНЫХ СЕРВИСОВ

Аннотация: В статье рассмотрены существующие системы для автоматизации развертывание сервисов

Ключевые слова: облачные сервисы, развертывание сервисов, Kubernetes,
Docker, Docker Swarm, Mesos, Rancher.

Farrakhov I.G.
gradute student
Naberezhnye Chelny Institute (branch) of KFU
Russia, Naberezhnye Chelny
Yakupov I.M.
gradute student
Naberezhnye Chelny Institute (branch) of KFU
Russia, Naberezhnye Chelny

AUTOMATED TOOLS FOR DEPLOYING CLOUD SERVICES

Abstract: The article discusses the existing systems for automating the deployment of services

Keywords: cloud services, service deployment, kubernetes, Docker, Docker Swarm, Mesos, Rancher.

С развитием интернета и технологий разработки
подход к разработке и предоставлению услуг кардинально изменился. Сегодня в интернете хранится большое количество веб-сайтов и разнообразной информации. Вместе с развитием интернета развивались и веб-сервисы, и способы их размещения на серверах. Изначально это были собственные серверы компаний, которые имели множество недостатков начиная от потребности хранить собственный штат сотрудников и заканчивая тем, что ресурсы каждого из серверов не использовались на 100%, позже

появились виртуальные машины, которые тоже имели свои недостатки в виде потребности виртуализировать для каждой виртуальной машины операционную систему и физические компоненты компьютера. Еще позже появились контейнеры, используемые на данный момент в мире. Относительно недавно количество контейнеров, которые разворачивает компания выросли к сотням, а иногда и тысячам и остро встал вопрос автоматического развертывание таких сервисов. Именно эту проблему и решают такие системы как Kubernetes. Они позволяют автоматически разворачивать множество микросервисов, настроив их один раз конфигурационными файлами, а также поддерживать их работу.

Docker – это технология виртуализации контейнеров. Она как очень легкая виртуальная машина – VM. Это приложение для построения контейнеров, работа которого на самом деле заключается в помощи людям в создании контейнеров и приложений внутри изолированного пространства, а также хранения копий этих контейнеров для того, чтобы потом делиться ими среди своих товарищей по команде и без проблем разворачивать такие копии как в облаке, так и на локальном компьютере разработчика для анализа, например, если возникли какие-то проблемы.

Есть несколько проблем, которые Docker решает. Первая из них о том, что VM является достаточно большим вычислительным ресурсом. Средняя виртуальная машина – это копия операционной системы, которая работает поверх гипервизора – программного обеспечение, позволяющее параллельно управлять несколькими операционными системами, запущенными поверх другой операционной системы, работающей поверх физического оборудования, над которым потом находится ваша программа. Это представляет определенные проблемы в отношении скорости и производительности, а также некоторые проблемы в ловкому среде, а также проблемы с быстродействием под время развертывания и остановки работы.

Kubernetes – это механизм организации контейнеров (COE) с открытым кодом, вдохновленный проектом Google под названием Borg. Kubernetes используется для организации групп контейнеров, представляющих экземпляры программ, которые отделены от машин, на которых они работают. Поскольку количество контейнеров в кластере увеличивается до сотен или тысяч экземпляров, а компоненты приложений разворачиваются как отдельные контейнеры, Kubernetes приходит на помощь, обеспечивая основу для развертывания, управления, автоматического масштабирования, высокой доступности и соответствующих задач.

Контейнеры – хороший способ объединить и запустить ваши программы. В производственной среде вам нужно управлять контейнерами, в которых запущены программы, и убедиться, что нет простоев. Например, если контейнер опускается, нужно запускать другой контейнер. Вот так на помощь приходит Kubernetes. Kubernetes предоставляет фреймворк для устойчивого запуска распределенных систем. Он заботится о масштабировании и

восстановлении после отказа для вашего приложения, предоставляет схемы развертывания и т. д. Например, Kubernetes может легко управлять развертыванием сервисов для вашей системы, или если была потеряна связь с одним из контейнеров, или даже компьютером, то Kubernetes знает, какие контейнеры и с какими параметрами были запущены на этом компьютере и за считанные секунды запустит все потерянные контейнеры на других компьютерах. Хотя данные из оперативной памяти контейнеров будут потеряны, но количество работающих сервисов не будет уменьшена и пользователю нужно лишь будет повторить свой вызов до сервиса чтобы получить ответ и весь этот процесс можно автоматизировать и только вмешиваться в него для увеличения контейнеров или для изучения результатов работы системы за определенное время для принятия определенных решений.

Docker Swarm – это альтернативный, собственный механизм блокировки оркестрации контейнеров, который координирует размещение и управление контейнерами между несколькими хостами Docker Engine. Docker Swarm позволяет вам общаться непосредственно с множеством контейнеров, вместо того, чтобы общаться с каждым Docker контейнером отдельно.

Docker compose – это простой, но мощный инструмент, который используется для запуска нескольких контейнеров в качестве одной службы. Например, предположим, что есть программа, которая требует Nginx как веб-сервера и PostgreSQL как службы баз данных. В этом случае с помощью docker-compose можно создать один единственный файл docker-compose.yml), который создаст оба контейнера в качестве одной службы. У созданного множества сервисов будет собственная внутренняя сеть, и много других преимуществ.

Apache Mesos, общекластерный менеджер ресурсов, широко применяется в нескольких облаках и центрах обработки данных. Mesos стремится обеспечить высокое использование кластеров с помощью тонкозернистого совместного планирования ресурсов и справедливости ресурсов среди нескольких пользователей благодаря распределению на основе доминирующей справедливости ресурсов.

DRF учитывает различные типы ресурсов (ЦП, память, дисковый ввод / вывод), что требуется каждой программой, и определяет долю каждого ресурса кластера, который может быть назначен приложениям. Месос принял двухуровневую политику планирования: DRF для распределения ресурсов на конкурирующие рамки и планирование уровня задач каждой структурой для ресурсов, выделенных на предыдущем шаге. Мы провели эксперименты в локальном кластере Mesos, когда они использовались с такими фреймворками, как Apache Aurora, Marathon и нашим собственным фреймворком Scylla, для изучения справедливости ресурсов и использования кластера.

Mesos объединяет все ресурсы в кластере и позволяет четко распределить ресурсы, позволяя и применяя несколько приложений (называемых Mesos framework) для совместного планирования своих задач на виртуальных машинах / узлах. Mesos использует DRF для распределения ресурсов в фреймворки, а затем фреймворки используют алгоритмы планирования для планирования задач в пределах выделенных ресурсов.

Использованные источники:

1. Diomidis S. Version Control Systems / Diomidis Spinellis, 2005.
2. Mojtaba S. Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices / Mojtaba Shahin, 2017.
3. Anderson C. Docker / Anderson Charles, 2015.
4. Saha P. Exploring the Fairness and Resource Distribution in an Apache Mesos Environment / Saha Pankaj. Cloud and Big Data Laboratory, State University of, 2019.
5. Stolberg S. Enabling Agile Testing Through Continuous Integration / Stolberg Sean, 2009