Д.А. Костюк, П.Н. Луцюк
Брест, Брестский государственный технический университет

Практическое изучение средств контейнерной виртуализации и платформы Kubernetes

Аннотация

Приводится опыт разработки учебного курса по основам использования оркестратора контейнеров Kubernetes, включая принципы контейнерной виртуализации, архитектурные особенности и ключевые компоненты платформы Kubernetes, развертывание и масштабирование приложений на ее основе. Приведена структура практической части курса, меры по обеспечению работоспособности изучаемого программного обеспечения в условиях изолированного сегмента сети и ограниченного Интернет-доступа, обсуждаются способы преодоления высокого входного порога Kubernetes.

Введение

Контейнеризация приложений — существенная часть актуального подхода к системной инженерии, предполагающего интеграцию разработки и эксплуатации программных систем при построении слож-

ных приложений. В основе лежит использование комплекса таких современных программных технологий, как виртуализация, автоматический аудит и контроль производительности, решение задач с применением гетерогенных систем, включающих несколько различных платформ.

Одним из характерных средств реализации данного подхода является Kubernetes — развиваемая Google сложнокомпонентная система оркестрации контейнеров с открытым исходным кодом.

Объекты Kubernetes разворачивают и масштабируют приложения на основе требований к памяти, ЦП и др. При этом поддерживается любая система контейнерной виртуализации, удовлетворяющая Open Container Initiative (OCI). Kubernetes обеспечивает масштабирование и балансировку нагрузки, автоматическое обнаружение сервисов и управление секретами. При этом контейнеры, являющиеся компонентами одного или нескольких приложений, изолированы друг от друга, если только разработчику/devops-инженеру не требуется обратного. Платформа берет на себя технические сложности по обеспечению прозрачного взаимодействия контейнеров и автообнаружения сервисов, предоставляя соответствующие возможности с помощью API.

Специфика изучаемого материала

Проблема, с которой регулярно сталкиваются при освоении Kubernetes — высокий порог вхождения. С одной стороны причина в достаточно большом наборе сущностей данной платформы, благодаря которым кластер на базе Kubernetes действует как единое целое, а с другой — в значительном числе входящих в ее состав компонентов, некоторые из которых к тому же являются заменяемыми. Универсальный кластер, автоматически выполняющий обновления, масштабирование и самовосстановление, демонстрирует достаточно высокую концептуальную сложность своего устройства. По этой причине, несмотря на высокую востребованность, преподавание данной платформы студентам часто оказывается проблематичным.

В нашем случае разрабатывался курс по изучению Kubernetes студентами старших курсов, уже знакомыми с рядом необходимых технологий, включая компьютерные сети, элементы системного администрирования GNU/Linux, архитектуры клиент-серверных приложений и др.

Разработанный учебный курс рассчитан на первую ступень высшего образования, и предполагает дальнейшее развитие в рамках второй ступени. Теоретический материал основан на официальной документации Kubernetes [1], а лабораторный практикум охватывает лишь часть теоретического материала, в результате чего студенты имеют дело с меньшим количеством объектов платформы.

Конфигурирование программного обеспечения и сетевой доступ

Практическая часть курса построена на основе Minikube — специализированного дистрибутива Kubernetes, предназначенного для развёртывания на локальной машине с совмещением аппаратной и контейнерной виртуализации [2]. Minikube имеет ряд ограничений, связанных в первую очередь с его локальной природой. Это не является проблемой, поскольку практическая часть курса в силу ограничений по времени не затрагивает ряд несовместимых аспектов платформы: такие темы как, например, реализация аффинитета и антиаффинитета, рассматриваются только в теоретической части.

Из внешних программных зависимостей требуется только VirtualBox, который доступен в стандартных репозиториях Linux; остальное скачивается из сети при установке и развёртывании:

- Minikube скачивает готовый установочный образ с Docker и компонентами Kubernetes, а также актуальные версии kubelet и kubeadm;
- далее при развёртывании приложений в кластере Kubernetes контейнеры приложения скачивает уже Docker.

В связи с этим генерируется сравнительно большой сетевой трафик (бинарные файлы minikube-linux-amd64 и kubectl, образ виртуальной машины minikube-v*.iso, а также образы контейнеров имеют общий объём более 600 Мб), и потому одной из задач, которые потребовалось решить для входящего в курс лабораторного практикума, было функционирование Minikube в изолированном сегменте локальной сети, в условиях ограниченного доступа к внешним Интернет-ресурсам (помимо уменьшения трафика и сопутствующих временных задержек, к такой схеме подталкивала внутренняя политика безопасности локальной сети, предполагающая персонализованный Интернет-доступ через VPN).

Было рассмотрено и опробовано два варианта локального использования Kuberbnetes:

1. создание приватного реестра Docker и перенастройка системы на его эксклюзивное использование, а также развёртывание приватного репозитория образов для Minikube;
2. запуск Minikube и пробное развёртывание учебных приложений при включённом доступе к Интернет с последующим клонированием полученного профиля на рабочие станции учебного класса (решаются вопросы как развёртывания образов для Minikube, так и импорт образов Docker через файловую систему вместо их получения из внешней сети).

После экспериментов с обоими вариантами мы остановились на втором как наименее трудоёмком, с учетом имеющейся действующей системы тиражирования образов для рабочих станций [3].

При этом следует отметить, что несмотря на стандартную политику обновления контейнеров Kubernetes `IfNotPresent` (скачивать образ контейнера в случае его отсутствия в локальном кэше), на практике потребовалось устанавливать ее значение `Never`, чтобы обеспечить полную воспроизводимость использования именно локальных образов без обновления.

Структура практикума

Структура разработанного практикума для студентов первой ступени включает 4 лабораторные работы. Первая работа носит ознакомительный характер: в ней рассказывается о Minikube и системах виртуализации, которые он может использовать на различных аппаратных платформах, а также об особенностях установки в случае использования на собственных устройствах. Во второй работе рассматриваются особенности доступа к кластеру Kubernetes с помощью командной строки (kubectl), включая минимальное конфигурирование доступа (kubectl proxy), и веб-интерфейса (Kubernetes Dashboard). В третьей работе выполняется развертывание минимального веб-приложения в кластере, включая проверку созданных Deployment, ReplicaSet и Подов, а также Service с доступом через NodePort. Четвёртая работа позволяет изучить развертывание многокомпонентного приложения с заготовленной клиентской частью на базе веб-технологий и серверной подсистемой на основе СУБД MongoDB.

Литература

[1] Kubernetes Documentation. **https://kubernetes.io/docs/home/**

[2] Hello Minikube.
https://kubernetes.io/docs/tutorials/hello-minikube/

[3] *Пойта П. С., Костюк Д. А., Дереченник С. С., Луцюк П. Н.* Повышение сетевой безопасности в компьютерном парке вуза за счет буферизации и изоляции ресурсов // Электроника инфо. – 2013. – No.6 (96). – С. 111–113.