

Амиму Кнаббену (Amim Knabben), Рикардо Кацу (Ricardo Katz), Мэту Фенвику (Matt Fenwick), Антонио Охеа (Antonio Ojea), Раджасу Какодару (Rajas Kakodar) и Микаэлю Клюзо (Mikael Cluseau) за многочасовые исследования K8s по ночам и выходным и увлекательные соревнования по крику. Эндрю Стойокосу (Andrew Stoycos), возглавлявшему группу политик в SIG Network. Моей жене и семье, позволившим мне писать эту книгу по субботам. Гари (Gary), Роне (Rona), Норе (Nora) и Джинджину (Gingin) за помощь моей маме.

– Джей

Кейт (Kate) и всем моим близким, поддерживавшим меня в этом путешествии. Спасибо команде LionKube, особенно Одри (Audrey) за организацию работы и Шарифу (Sharif) за помощь и поддержку. Также моему соавтору Джею (Jay), предложившему мне принять участие в работе над этой книгой вместе с ним, я благодарю тебя за это! Без твоей целеустремленности и упорства у нас ничего не получилось бы.

– Крис

Оглавление

1	■ Почему появился Kubernetes.....	24
2	■ Зачем нужны модули Pod?.....	40
3	■ Создание модулей Pod.....	68
4	■ Использование контрольных групп для управления процессами в модулях Pod.....	103
5	■ Интерфейсы CNI и настройка сети в модулях Pod.....	132
6	■ Устранение проблем в крупномасштабных сетях.....	154
7	■ Хранилища в модулях Pod и CSI.....	179
8	■ Реализация и моделирование хранилищ.....	198
9	■ Запуск модулей Pod: как работает kubelet.....	221
10	■ DNS в Kubernetes.....	243
11	■ Плоскость управления.....	257
12	■ etcd и плоскость управления.....	272
13	■ Безопасность контейнеров и модулей Pod.....	296
14	■ Безопасность узлов и Kubernetes.....	312
15	■ Установка приложений.....	343

Содержание

Оглавление	6
Предисловие	14
Благодарности	15
О книге	17
Об авторах	21
Об иллюстрации на обложке	23

1 Почему появился Kubernetes	24
1.1 Предварительный обзор некоторых основных терминов	25
1.2 Проблема дрейфа инфраструктуры и Kubernetes	26
1.3 Контейнеры и образы	27
1.4 Базовая основа Kubernetes	29
1.4.1 Все инфраструктурные правила в Kubernetes определяются в обычных файлах YAML	31
1.5 Возможности Kubernetes	32
1.6 Компоненты и архитектура Kubernetes	34
1.6.1 Kubernetes API	35
1.6.2 Пример первый: интернет-магазин	37
1.6.3 Пример второй: онлайн-решение для благотворительности	37
1.7 Когда не стоит использовать Kubernetes	38
Итоги	38

2 Зачем нужны модули Pod?	40
2.1 Пример веб-приложения	42
2.1.1 Инфраструктура нашего веб-приложения	44
2.1.2 Эксплуатационные требования	45
2.2 Что такое Pod?	46
2.2.1 Пространства имен в Linux	47
2.2.2 Kubernetes, инфраструктура и Pod	49
2.2.3 Объект Node	51
2.2.4 Наше веб-приложение и плоскость управления	55
2.3 Создание веб-приложения с помощью kubectl	56
2.3.1 Сервер Kubernetes API: kube-apiserver	57
2.3.2 Планировщик Kubernetes: kube-scheduler	58

2.3.3	Контроллеры инфраструктуры	59
2.4	Масштабирование, высокодоступные приложения и плоскость управления	63
2.4.1	Автоматическое масштабирование	65
2.4.2	Управление затратами	66
Итоги	67

3	Создание модулей Pod	68
3.1	Общий обзор примитивов Kubernetes	71
3.2	Что такое примитивы Linux?	72
3.2.1	Примитивы Linux – это инструменты управления ресурсами	73
3.2.2	Все сущее является файлом (или файловым дескриптором)	74
3.2.3	Файлы можно комбинировать	75
3.2.4	Настройка kind	76
3.3	Использование примитивов Linux в Kubernetes	78
3.3.1	Предварительные условия для запуска модуля Pod	78
3.3.2	Запуск простого модуля Pod	79
3.3.3	Исследование зависимостей модуля Pod от Linux	81
3.4	Создание модуля Pod с нуля	86
3.4.1	Создание изолированного процесса с помощью chroot	87
3.4.2	Использование mount для передачи данных процессам	89
3.4.3	Защита процесса с помощью unshare	91
3.4.4	Создание сетевого пространства имен	92
3.4.5	Проверка работоспособности процесса	93
3.4.6	Ограничение потребления процессора с помощью cgroups	94
3.4.7	Создание раздела resources	95
3.5	Использование модуля Pod в реальном мире	96
3.5.1	Проблема сети	97
3.5.2	Как kube-проху реализует сервисы Kubernetes с помощью iptables	98
3.5.3	Использование модуля kube-dns	98
3.5.4	Другие проблемы	100
Итоги	102

4	Использование контрольных групп для управления процессами в модулях Pod	103
4.1	Модули Pod простаивают до завершения подготовительных операций	104
4.2	Процессы и потоки в Linux	106
4.2.1	Процессы systemd и init	108
4.2.2	Контрольные группы для процессов	109
4.2.3	Реализация контрольных групп для обычного модуля Pod	112
4.3	Тестирование контрольных групп	114
4.4	Как kubelet управляет контрольными группами	115
4.5	Как kubelet управляет ресурсами	116
4.5.1	Почему ОС не может использовать подкачку в Kubernetes?	117
4.5.2	Хак: настройка приоритета «для бедных»	118
4.5.3	Хак: настройка HugePages с помощью контейнеров инициализации	119
4.5.4	Классы QoS: почему они важны и как они работают	120
4.5.5	Создание классов QoS путем настройки ресурсов	121

4.6	Мониторинг ядра Linux с помощью Prometheus, cAdvisor и сервера API.....	122
4.6.1	Публикация метрик обходится недорого и имеет большую ценность	124
4.6.2	Почему Prometheus?	125
4.6.3	Создание локального сервиса мониторинга Prometheus	126
4.6.4	Исследование простоев в Prometheus.....	129
Итоги	131

5 Интерфейсы CNI и настройка сети в модулях Pod..... 132

5.1	Зачем нужны программно-определяемые сети в Kubernetes	134
5.2	Реализация Kubernetes SDN на стороне сервиса: kube-proxy.....	136
5.2.1	Плоскость данных в kube-proxy	138
5.2.2	Подробнее о NodePort	140
5.3	Провайдеры CNI	141
5.4	Два плагина CNI: Calico и Antrea	143
5.4.1	Архитектура плагинов CNI	143
5.4.2	Давайте поэкспериментируем с некоторыми CNI	144
5.4.3	Установка провайдера CNI Calico	146
5.4.4	Организация сети в Kubernetes с OVS и Antrea.....	149
5.4.5	Замечание о провайдерах CNI и kube-proxy в разных ОС	152
Итоги	153

6 Устранение проблем в крупномасштабных сетях..... 154

6.1	Sonobuou: инструмент подтверждения работоспособности кластера.....	155
6.1.1	Трассировка движения данных модулей Pod в кластере	156
6.1.2	Настройка кластера с CNI-провайдером Antrea	157
6.2	Исследование особенностей маршрутизации в разных провайдерах CNI с помощью команд <code>arp</code> и <code>ip</code>	158
6.2.1	Что такое IP-туннель и почему его используют провайдеры CNI?	159
6.2.2	Сколько пакетов проходит через сетевые интерфейсы CNI?.....	160
6.2.3	Маршруты	161
6.2.4	Инструменты для CNI: Open vSwitch (OVS).....	163
6.2.5	Трассировка движения данных активных контейнеров с помощью <code>tcpdump</code>	164
6.3	kube-proxy и iptables.....	166
6.3.1	iptables-save и diff.....	166
6.3.2	Как сетевые политики изменяют правила CNI	167
6.3.3	Как реализуются политики?.....	170
6.4	Входные контроллеры.....	172
6.4.1	Настройка Contour и кластера kind для изучения входных контроллеров	173
6.4.2	Настройка простого модуля Pod с веб-сервером	174
Итоги	178

7	Хранилища в модулях Pod и CSI	179
7.1	Небольшое отступление: виртуальная файловая система (VFS) в Linux	181
7.2	Три вида хранилищ для Kubernetes	182
7.3	Создание PVC в кластере kind.....	184
7.4	Интерфейс контейнерного хранилища (CSI)	188
7.4.1	Проблема внутреннего провайдера	189
7.4.2	CSI как спецификация, работающая внутри Kubernetes	191
7.4.3	CSI: как работает драйвер хранилища	193
7.4.4	Привязка точек монтирования.....	193
7.5	Краткий обзор действующих драйверов CSI.....	194
7.5.1	Контроллер	194
7.5.2	Интерфейс узла	195
7.5.3	CSI в операционных системах, отличных от Linux	196
	Итоги	196
8	Реализация и моделирование хранилищ	198
8.1	Микрокосм в экосистеме Kubernetes: динамическое хранилище	199
8.1.1	Оперативное управление хранилищем: динамическое выделение ресурсов	200
8.1.2	Локальное хранилище в сравнении с emptyDir	201
8.1.3	Тома PersistentVolume	203
8.1.4	Интерфейс контейнерного хранилища (CSI)	204
8.2	Динамическая подготовка выигрывает от CSI, но не зависит от него	205
8.2.1	Классы хранилищ (StorageClass)	206
8.2.2	Вернемся к центрам обработки данных.....	207
8.3	Варианты организации хранилищ в Kubernetes	209
8.3.1	Секреты: эфемерная передача файлов	209
8.4	Как выглядит типичный провайдер динамического хранилища?	212
8.5	hostPath для управления системой и/или доступа к данным	214
8.5.1	hostPath, CSI и CNI: канонический вариант использования	214
8.5.2	Cassandra: пример реального хранилища в Kubernetes	217
8.5.3	Дополнительные возможности и модель хранения в Kubernetes.....	218
8.6	Дополнительная литература.....	219
	Итоги	220
9	Запуск модулей Pod: как работает kubelet	221
9.1	kubelet и узел	222
9.2	Основы kubelet.....	223
9.2.1	Среда выполнения контейнеров: стандарты и соглашения	224
9.2.2	Конфигурационные параметры и API агента kubelet.....	225
9.3	Создание модуля Pod и его мониторинг.....	228
9.3.1	Запуск kubelet	229
9.3.2	После запуска: жизненный цикл узла	230
9.3.3	Механизм аренды и блокировки в etcd, эволюция аренды узла	230
9.3.4	Управление жизненным циклом Pod в kubelet.....	231
9.3.5	CRI, контейнеры и образы: как они связаны	233
9.3.6	kubelet не запускает контейнеры: это делает CRI.....	233
9.3.7	Приостановленный контейнер: момент истины	235

9.4	Интерфейс времени выполнения контейнеров (CRI).....	235
9.4.1	Сообщаем Kubernetes, где находится среда выполнения контейнеров.....	235
9.4.2	Процедуры CRI.....	236
9.4.3	Абстракция kubelet вокруг CRI: GenericRuntimeManager.....	236
9.4.4	Как вызывается CRI?.....	237
9.5	Интерфейсы kubelet.....	237
9.5.1	Внутренний интерфейс среды выполнения.....	237
9.5.2	Как kubelet извлекает образы: интерфейс ImageService.....	239
9.5.3	Передача ImagePullSecret в kubelet.....	240
9.6	Дополнительная литература.....	241
Итоги	241

10	DNS в Kubernetes	243
10.1	Краткое введение в DNS (и CoreDNS).....	243
10.1.1	NXDOMAIN, записи A и записи CNAME.....	244
10.1.2	Модулям Pod нужен внутренний DNS.....	246
10.2	Почему StatefulSet, а не Deployment?.....	248
10.2.1	DNS и автономные сервисы.....	248
10.2.2	Постоянные записи DNS в StatefulSet.....	250
10.2.3	Развертывание с несколькими пространствами имен для изучения свойств модуля DNS.....	250
10.3	Файл resolv.conf.....	252
10.3.1	Краткое примечание о маршрутизации.....	252
10.3.2	CoreDNS: вышестоящий сервер имен для ClusterFirst DNS.....	254
10.3.3	Разбор конфигурации плагина CoreDNS.....	255
Итоги	256

11	Плоскость управления	257
11.1	Плоскость управления.....	258
11.2	Особенности сервера API.....	259
11.2.1	Объекты API и пользовательские ресурсы.....	259
11.2.2	Определения пользовательских ресурсов (CRD).....	261
11.2.3	Планировщик.....	261
11.2.4	Краткий обзор фреймворка планирования.....	267
11.3	Диспетчер контроллеров.....	267
11.3.1	Хранилище.....	268
11.3.2	Учетные данные сервисов и токены.....	269
11.4	Облачные диспетчеры контроллеров Kubernetes (CCM).....	269
11.5	Дополнительная литература.....	271
Итоги	271

12	etcd и плоскость управления	272
12.1	Заметки для нетерпеливых.....	273
12.1.1	Мониторинг производительности etcd с помощью Prometheus.....	274
12.1.2	Когда нужно настраивать etcd.....	278
12.1.3	Пример: быстрая проверка работоспособности etcd.....	280
12.1.4	etcd v3 и v2.....	280
12.2	etcd как хранилище данных.....	281
12.2.1	Можно ли запустить Kubernetes в других базах данных?.....	281

12.2.2	Строгая согласованность	283
12.2.3	Согласованность в etcd обеспечивают операции fsync	283
12.3	Обзор интерфейса Kubernetes с etcd	285
12.4	Задача etcd – надежное хранение фактов.....	285
12.4.1	Журнал упреждающей записи etcd	287
12.4.2	Влияние на Kubernetes	287
12.5	Теорема CAP.....	287
12.6	Балансировка нагрузки на уровне клиента и etcd	289
12.6.1	Ограничения по размеру: о чем (не) следует беспокоиться	289
12.7	Шифрование хранимых данных в etcd	291
12.8	Производительность и отказоустойчивость etcd в глобальном масштабе	292
12.9	Интервал отправки контрольных сообщений в высокораспределенной etcd	292
12.10	Настройка клиента etcd в кластере kind	293
12.10.1	Запуск etcd в окружении, отличном от Linux.....	294
Итоги	295

13 Безопасность контейнеров и модулей Pod..... 296

13.1	Радиус взрыва	297
13.1.1	Уязвимости	298
13.1.2	Вторжение	298
13.2	Безопасность контейнера	298
13.2.1	Планирование обновления контейнеров и пользовательского программного обеспечения	299
13.2.2	Контроль контейнеров	299
13.2.3	Пользователи в контейнерах – не запускайте ПО от имени root	300
13.2.4	Используйте наименьшие возможные контейнеры.....	300
13.2.5	Происхождение контейнера	301
13.2.6	Линтеры для контейнеров	302
13.3	Безопасность модулей Pod.....	302
13.3.1	Контекст безопасности	303
13.3.2	Расширение привилегий и возможностей	305
13.3.3	Политики безопасности Pod (PSP)	307
13.3.4	Не внедряйте автоматически токен учетной записи сервиса	309
13.3.5	Модули Pod с привилегиями root	309
13.3.6	Граница безопасности	310
Итоги	311

14 Безопасность узлов и Kubernetes

14.1	Безопасность узла.....	312
14.1.1	Сертификаты TLS.....	313
14.1.2	Неизменяемые ОС и применение исправлений на узлах	314
14.1.3	Изолированные среды выполнения контейнеров	315
14.1.4	Атаки на ресурсы	316
14.1.5	Единицы измерения потребления процессора	317
14.1.6	Единицы измерения объема памяти	317
14.1.7	Единицы измерения объема хранилища	318
14.1.8	Сети хостов и модулей Pod	318
14.1.9	Пример модуля Pod	319

14.2	Безопасность сервера API	320
14.2.1	Управление доступом на основе ролей (RBAC)	320
14.2.2	Определение RBAC API	321
14.2.3	Ресурсы и подресурсы	323
14.2.4	Субъекты и RBAC.....	325
14.2.5	Отладка RBAC.....	326
14.3	Authn, Authz и Secret	326
14.3.1	Учетные записи сервисов IAM: защита облачных API	327
14.3.2	Доступ к облачным ресурсам	328
14.3.3	Частные серверы API	329
14.4	Безопасность сети	329
14.4.1	Сетевые политики.....	330
14.4.2	Балансировщики нагрузки	334
14.4.3	Агент открытой политики (OPA)	335
14.4.4	Коллективная аренда.....	338
14.5	Советы по Kubernetes	341
Итоги	341

15 Установка приложений

15.1	Размышления о приложениях в Kubernetes	344
15.1.1	Масштаб приложения влияет на выбор инструментов	345
15.2	Приложения на основе микросервисов обычно требуют тысячи строк определения конфигурации	345
15.3	Переосмысление установки приложения Guestbook в реальных условиях	346
15.4	Установка набора инструментов Carvel.....	347
15.4.1	Часть 1: разделение ресурсов на отдельные файлы	347
15.4.2	Часть 2: исправление файлов приложения с помощью ytt.....	349
15.4.3	Часть 3: развертывание приложения Guestbook и управление им	351
15.4.4	Часть 4: создание оператора kapp для упаковки приложения и управления им.....	355
15.5	И снова об операторах Kubernetes	359
15.6	Tanzu Community Edition: пример комплексного набора инструментов Carvel	362
Итоги	363

Предметный указатель.....	365
---------------------------	-----

Предисловие

Мы написали эту книгу для всех, кто хочет получить новые знания о K8s (Kubernetes) и сразу же углубиться в различные темы, связанные с хранением, сетевыми взаимодействиями и использованием разнообразных инструментов.

Мы не ставили перед собой цель написать исчерпывающее руководство по всем возможностям K8s API (это просто невозможно), но искренне верим, что, прочитав эту книгу, пользователи обретут понимание, которое поможет им по-новому взглянуть на сложные задачи организации инфраструктуры в промышленных кластерах и увидеть общее развитие ландшафта Kubernetes в более широком контексте.

Конечно, есть немало книг, позволяющих пользователям изучить основы Kubernetes, но мы хотели написать книгу, рассказывающую об основных технологиях, составляющих Kubernetes. Здесь мы расскажем вам все тонкости организации сети и плоскости управления, а также некоторые другие темы, чтобы вы смогли понять внутренние особенности работы Kubernetes. Понимание этих деталей сделает вас лучшим инженером DevOps и программистом.

Мы также надеемся вдохновить новых пользователей Kubernetes. Свяжитесь с нами в Твиттере (@jayunit100, @chrislovcnm), если решите принять участие в жизни сообщества Kubernetes или помочь нам добавить больше примеров для этой книги.

Благодарности

Мы хотим поблагодарить сообщество и компании, поддерживающие Kubernetes. Без них и их постоянных усилий не было бы этого замечательного программного обеспечения. Мы хотели бы упомянуть всех причастных, но боимся, что кого-то упустим, поэтому извиняемся заранее.

Спасибо нашим друзьям и наставникам в SIG Network (Микаэлю Клузо (Mikael Cluseau), Халеду Хендиаку (Khaled Hendiak), Тиму Хокинсу (Tim Hockins), Антонио Охеа (Antonio Ojea), Рикардо Кацу (Ricardo Katz), Мэтту Фенвику (Matt Fenwick), Дэну Уиншипу (Dan Winship), Дэну Уильямсу (Dan Williams), Кейси Календеро (Casey Calendero), Кейси Девенпорт (Casey Davenport), Эндрю Си (Andrew Sy) и многим, многим другим); неутомимым разработчикам открытого исходного кода в сообществах SIG Network и SIG Windows (Марку Розетти (Mark Rosetti), Джеймсу Стареванту (James Sturevant), Клаудио Белу (Claudio Belu), Амиму Кнаббену (Amim Knabben)); первым основателям Kubernetes (Джо Беду (Joe Beda), Брендану Барнсу (Brendan Burns), Вилле Айкасу (Ville Aikas) и Крейгу Маклаки (Craig McLuckie)); а также инженерам из Google, включая Брайана Гранта (Brian Grant) и Тима Хокина (Tim Hockin), присоединившимся к ним вскоре после этого.

Мы благодарны духовным лидерам сообщества: Тиму Сент-Клеру (Tim St. Clair), Джордану Лиггиту (Jordan Liggitt), Бриджит Кромхаут (Bridget Kromhaut) и многим другим. Мы также хотели бы поблагодарить Раджаса Какодара (Rajas Kakodar), Анушу Хедж (Anusha Hedge) и Неху Лохию (Neha Lohia) за создание потрясающей команды SIG Network India, внесших колоссальное количество предложений, которые мы надеемся учесть в следующем издании (или возможном продолжении) этой книги, где мы предполагаем подробнее рассказать о приемах организации сети и о прокси-сервере kube-проху.

Джей также хотел бы поблагодарить Клинта Китсона (Clint Kitson) и Аарти Ганесана (Aarthi Ganesan), давших возможность работать над этой книгой, будучи сотрудником VMware, а также его коллег в VMware (Амима (Amim) и Зака (Zac)), постоянно остававшихся рядом и по-

могавших советами. И конечно же, спасибо Фрэнсису Бурану (Frances Buran), Карен Миллер (Karen Miller) и многим другим сотрудникам Manning Publications, помогавшим готовить эту книгу к печати.

Наконец, спасибо всем нашим рецензентам: Алу Кринкеру (Al Krinker), Алессандро Кампейсу (Alessandro Campeis), Александру Эрциу (Alexandru Herciu), Аманде Деблер (Amanda Debler), Андреа Косентино (Andrea Cosentino), Андреесе Сакко (Andres Sacco), Анупаму Сенгупте (Anupam Sengupta), Бену Фенвику (Ben Fenwick), Борко Джурковичу (Borko Djurkovic), Дарье Василенко (Daria Vasilenko), Элиасу Рангелю (Elias Rangel), Эрике Хоулу (Eric Hole), Эриксу Зеленке (Eriks Zelenka), Ойгену Кокалеа (Eugen Cocalea), Ганди Раджану (Gandhi Rajan), Ирине Романенко (Iryna Romanenko), Джареду Дункану (Jared Duncan), Джеффу Лиму (Jeff Lim), Джиму Амрайну (Jim Amrhein), Хуану Хосе Дурильо Баррионуэво (Juan José Durillo Barrionuevo), Мэту Фенвику (Matt Fenwick), Мэтту Велке (Matt Welke), Михалю Рутке (Michał Rutka), Риккардо Маротти (Riccardo Marotti), Робу Пачеко (Rob Pacheco), Робу Рютчу (Rob Ruetsch), Роману Левченко (Roman Levchenko), Райану Бартлетту (Ryan Bartlett), Убальдо Пескаторе (Ubaldo Pesatore) и Уэсли Рольнику (Wesley Rolnick). Ваши предложения помогли сделать эту книгу лучше.

О книге

Кому адресована эта книга

Всем желающим узнать больше о внутреннем устройстве Kubernetes, о том, как рассуждать о его режимах отказа и возможности расширения под нужды пользователей. Если вы не знаете, что такое Pod, то, конечно, можете приобрести эту книгу, но прежде прочтите какую-нибудь другую книгу, где вы сможете поближе познакомиться с терминологией.

Также книга пригодится операторам, желающим общаться на едином языке с сотрудниками ИТ-отделов, техническими директорами и другими руководителями о том, как внедрить Kubernetes, сохранив при этом основные принципы построения инфраструктуры, существовавшие до появления контейнеров. Эта книга действительно поможет преодолеть разрыв между новыми и старыми решениями по проектированию инфраструктуры. По крайней мере, мы на это надеемся!

Организация книги

Эта книга состоит из 15 глав:

- глава 1. Дает общий обзор Kubernetes для новичков;
- глава 2. Рассматривает идею модуля Pod как атомарного строительного блока для приложений и закладывает основы для последующих глав, подробно рассматривающих низкоуровневые детали Linux;
- глава 3. Углубляется в детали использования в Kubernetes низкоуровневых примитивов Linux для реализации концепций более высокого уровня, включая модули Pod;
- глава 4. Здесь начинается изучение внутренних особенностей процессов и их изоляции в Linux, которые являются одними из менее известных деталей ландшафта Kubernetes;
- глава 5. После подробного знакомства с модулями Pod углубляется в организацию сетевых взаимодействий между ними и рассказывает, как они соединяются друг с другом через разные узлы;

- глава 6. Вторая глава, посвященная сетевым взаимодействиям, описывающая более широкие аспекты работы модулей Pod и прокси-сервера (kube-проху), а также приемы их настройки и сопровождения;
- глава 7. Первая глава, посвященная вопросам организации хранилища. Здесь дается широкое введение в теоретические основы хранилищ Kubernetes, контейнерный интерфейс хранилища (Container Storage Interface, CSI) и его взаимодействия с kubelet;
- глава 8. Вторая глава, посвященная вопросам организации хранилища. Здесь рассматриваются некоторые более практические аспекты хранения данных, включая особенности emptyDir, Secrets и PersistentVolumes/Dynamic storage;
- глава 9. Углубляется в kubelet и рассматривает некоторые детали запуска модулей Pod и управления ими, включая такие понятия, как CRI, жизненный цикл узла и ImagePullSecrets;
- глава 10. DNS в Kubernetes – сложный механизм, используемый почти всеми контейнерными приложениями для локального доступа к внутренним сервисам. Здесь рассматривается CoreDNS – реализация сервиса DNS для Kubernetes – и порядок выполнения DNS-запросов разными модулями Pod;
- глава 11. Подробно обсуждает плоскость управления, упоминавшуюся в предыдущих главах, включая тонкости работы планировщика, диспетчера контроллеров и сервера API. Они образуют «мозг» Kubernetes и объединяют вместе все низкоуровневые концепции, обсуждавшиеся в предыдущих главах;
- глава 12. После обсуждения логики плоскости управления мы углубимся в etcd, надежный механизм консенсуса для Kubernetes, и особенности его настройки для удовлетворения потребностей плоскости управления Kubernetes;
- глава 13. Представляет обзор NetworkPolicies, RBAC и безопасности на уровне модулей Pod и узлов, о которых должен знать каждый администратор. В этой главе также обсуждается общее развитие политик безопасности модулей Pod;
- глава 14. Здесь рассматривается настройка безопасности на уровне узла и облака, а также другие аспекты безопасности Kubernetes, ориентированные на инфраструктуру;
- глава 15. Эта заключительная глава дает общий обзор прикладных инструментов на примере Carvel, набора инструментов для управления файлами YAML, создания приложений и долгосрочного управления жизненным циклом приложений.

О примерах программного кода

Для этой книги мы подготовили несколько примеров, которые вы найдете в репозитории GitHub (<https://github.com/jayunit100/k8sprototypes/>), в том числе примеры:

- использования `kind` для настройки реалистичной сети в локальных кластерах с помощью Calico, Antrea или Cilium;
- анализа метрик Prometheus в реальном мире;
- создания приложений с помощью набора инструментов Carvel;
- различных экспериментов, связанных с RBAC.

Эта книга также содержит множество примеров программного кода. Они оформлены шрифтом фиксированной ширины, чтобы вам было проще отличать его от основного текста.

Во многих случаях исходный код переформатирован, чтобы уместить его по ширине книжной страницы. В частности, мы добавили разрывы строк и отступы. В редких случаях даже этого было недостаточно, и мы добавили маркеры продолжения строки (↪). Многие листинги сопровождаются комментариями в тексте, подчеркивающими важные понятия. Получить выполняемые фрагменты кода можно из электронной версии книги по адресу <https://livebook.manning.com/book/core-kubernetes> и в репозитории GitHub <https://github.com/jayunit100/k8sprototypes/>.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

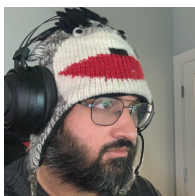
Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Manning Publications очень серьезно от-

носятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторах



Джей Вьяс, д-р наук (Jay Vyas, PhD), в настоящее время – штатный инженер в VMware. Работал над несколькими коммерческими дистрибутивами и платформами Kubernetes с открытым исходным кодом, включая OpenShift, VMware Tanzu, внутренними коллективными платформами Black Duck для Kubernetes и установкой Kubernetes для клиентов его

консалтинговой компании Rocket Rudolf, LLC. В течение нескольких лет был членом комитета по управлению проектами (Project Management Committee, PMC) в Apache Software Foundation, где работал над несколькими проектами в области больших данных. Он был связан с Kubernetes на различных должностях с момента его создания и в настоящее время уделяет большое внимание сообществам SIG-Windows и SIG-Network. Начинал с создания распределенных систем, одновременно защитив докторскую диссертацию по витринам данных в сфере биоинформатики (объединявшим базы данных в платформы для анализа человеческих и вирусных белковых карт – протеомов). Это привело его в мир больших данных и масштабируемых систем обработки данных и, наконец, в Kubernetes.

Связаться с Джейем можно по адресу [@jayunit100](https://twitter.com/jayunit100) в Твиттере, если вы заинтересованы в сотрудничестве... по какой угодно теме. Уделяет большое внимание спорту, ежедневно пробегает одну милю в спринтерском темпе и подтягивается до отказа. Также увлекается музыкой и имеет несколько синтезаторов, в том числе Prophet-6, который звучит как космический корабль.



Крис Лав (Chris Love) – сертифицированный сотрудник Google Cloud и соучредитель Lionkube. Больше 25 лет занимался разработкой программного обеспечения в таких компаниях, как Google, Oracle, VMware, Cisco, Johnson & Johnson и др. Как идейный лидер в Kubernetes и в сообществе DevOps, Крис Лав участвовал во многих проектах с открытым исход-

ным кодом, включая Kubernetes, kops (в должности руководителя AWS SIG), Bazel (внес вклад в разработку правил для Kubernetes) и Terraform (один из первых разработчиков плагина VMware). В число его профессиональных интересов входят: трансформация ИТ-культуры, технологии контейнеризации, методы и средства автоматизированного тестирования, Kubernetes, Golang (он же Go) и другие языки программирования. Лав обожает заниматься популяризацией DevOps, Kubernetes и технологий, а также обучать людей в сфере ИТ и программного обеспечения.

Вне работы любит кататься на лыжах, играть в волейбол, заниматься йогой и участвовать в мероприятиях на свежем воздухе. Кроме того, вот уже 20 лет занимается боевыми искусствами.

Если вы решите пообщаться с Крисом и задать ему свои вопросы, то сможете связаться с ним по адресу @chrislovenm в Twitter или LinkedIn.

Об иллюстрации на обложке

Изображение на обложке книги называется «Штерн, матрос у руля корабля». Оно взято с гравюры картины Альфредо Луксоро (Alfredo Luxoro), опубликованной в журнале «L'Illustrazione Italiana», № 19, от 9 мая 1880 года.

В те дни по одежде было легко определить, где живет человек, чем занимается и какое положение занимает в обществе. Мы в издательстве Manning славим изобретательность, предприимчивость и радость компьютерного бизнеса обложками книг, изображающими богатство региональных различий многовековой давности, оживших благодаря иллюстрациям, таким как эта.

Почему появился Kubernetes



В этой главе:

- почему появился Kubernetes;
- основные термины Kubernetes;
- конкретные примеры использования Kubernetes;
- высокоуровневые функции Kubernetes;
- когда нежелательно использовать Kubernetes.

Kubernetes – это платформа с открытым исходным кодом для размещения контейнеров и определения прикладных API для управления облачной семантикой обеспечения этих контейнеров хранилищами данных, сетевыми услугами, поддержкой безопасности и другими ресурсами. Kubernetes обеспечивает непрерывную синхронизацию всего пространства состояний ваших приложений, в том числе способов доступа к ним из внешнего мира.

Зачем внедрять Kubernetes в свое окружение? Не проще ли выделить все необходимые ресурсы вручную с помощью инструмента управления инфраструктурой, связанного с DevOps? Ответ зависит от того, как мы определяем процесс DevOps и его интеграцию в общий жизненный цикл приложения. DevOps продолжает расширяться и включает в себя процессы, инженеров и инструменты, которые поддерживают более автоматизированное администрирование приложений в центре обработки данных. Одно из условий успешного

Конец ознакомительного фрагмента.

Приобрести книгу можно

в интернет-магазине

«Электронный универс»

e-Univers.ru