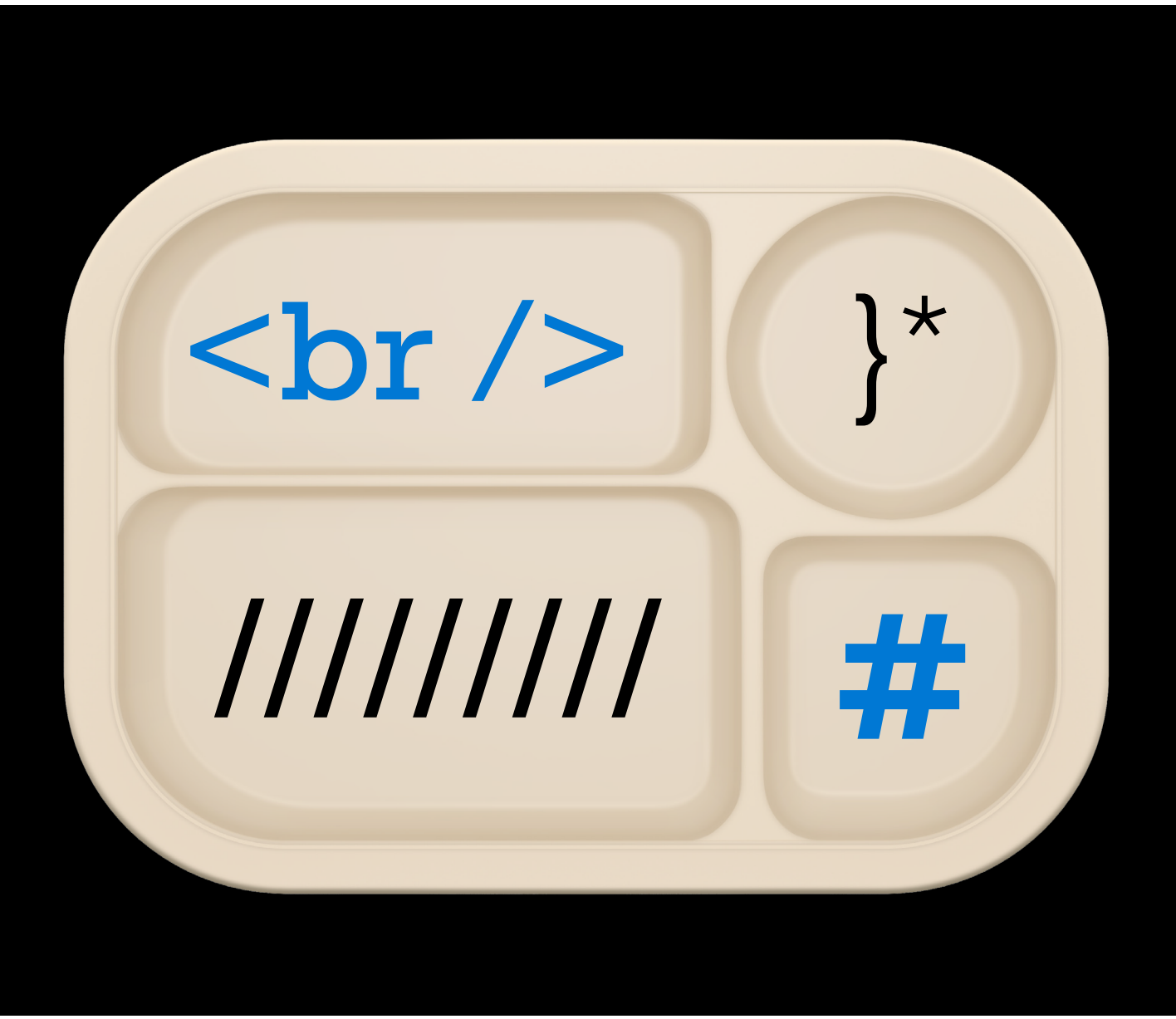
НАУЧИТЕСЬ РАБОТАТЬ С AZURE ЗА МЕСЯЦ

Второе издание
Иэн Фоулдс (Iain Foulds)



startHere(Azure);

Изучите 21 урок по Azure

Получите базовые знания об Azure, воспользовавшись уроками в этой книге. Зарегистрируйте бесплатную учетную запись Azure и используйте 200 долл. США на счете для выполнения упражнений. Продолжайте пользоваться своей учетной записью и получите популярные бесплатные сервисы на 12 месяцев и более 25 всегда бесплатных сервисов.

Отзывы о первом издании

Из первого издания книги *Научитесь работать с Azure за месяц* Иэна Фоулдса (Iain Foulds):

«Невероятно насыщенная книга для изучения базовых и углубленных концепций Azure всего за месяц!»,

— Сушил Шарма (Sushil Sharma), Galvanize

«Microsoft Azure быстро становится лидером в сфере публичного облака. Упражнения в этой книге помогут вам быстро освоить эту технологию»,

— Майкл Брайт (Michael Bright), специалист по поддержке разработчиков, внештатный консультант

«Прекрасное введение в Azure со множеством практических примеров. Охватывает широкий круг актуальных тем»,

— Свен Штумпф (Sven Stumpf), ING-DiBa AG

«Azure — океан возможностей. Эта книга держит вас на плаву, позволяя эффективно учиться на практических упражнениях и примерах»,

— Роман Левченко, Microsoft MVP

«Все, что нужно занятому разработчику, чтобы работать в Azure»,

— Роб Лорангер (Rob Loranger), внештатный разработчик

«Отличный способ понять широту предложений Azure на коротких практичных уроках»,

— Дэйв Корун (Dave Corun), Avanade

«Самая полная книга по Azure, которую я нашел, чтобы начать разрабатывать свои учебные проекты!»,

— Марко Джузеппе Салафия (Marco Giuseppe Salafia), аспирант, Университет Катании

«Это самая полезная книга о платформе Azure. Она хорошо организована, содержит тщательно подобранные и комплексные сведения. Все начинается с основ, и читатель учится создавать усложняющиеся конфигурации с помощью платформы Azure, чтобы обеспечить масштабируемость, высокую производительность и избыточность для размещенных приложений и сервисов. Эта книга может быть и учебным пособием для начинающих, и справочником для опытных пользователей»,

— Роберт Уолш (Robert Walsh), Excalibur Solutions

Научитесь работать с Azure за месяц

Второе издание

ИЭН ФОУЛДС (IAIN FOULDS)

MANNING
Шелтер-Айленд

Для получения информации и заказа этой и других книг издательства Manning через Интернет посетите сайт www.manning.com. При заказе этой книги в большом количестве издатель предлагает скидки. Для получения дополнительной информации обратитесь по следующему адресу:

Special Sales Department
Manning Publications Co.
20 Baldwin Road
PO Box 761
Shelter Island, NY 11964
Электронная почта: orders@manning.com

Manning Publications Co.
20 Baldwin Road
PO Box 761
Shelter Island, NY 11964

Редактор издательства:	Майк Стивенс (Mike Stephens)
Редактор-консультант по аудитории:	Фрэнсис Лефковиц (Frances Lefkowitz)
Редактор по технической разработке:	Карстен Стрёбек (Karsten Strøbaek)
Редактор-рецензент:	Александр Драгосавлевич (Aleksandar Dragosavljevic)
Выпускающий редактор:	Энтони Калькара (Anthony Calcara)
Графический редактор:	Дженнифер Хоул (Jennifer Houle)
Редактор-корректор:	Кэти Симпсон (Kathy Simpson)
Корректор:	Кэти Теннант (Katie Tennant)
Технический корректор:	Карстен Стрёбек (Karsten Strøbaek)
Наборщик:	Мария Тюдор (Marija Tudor)
Дизайнер обложки:	Лесли Хаимс (Leslie Haimes)

ISBN 9781617297625
Опубликовано в США

Посвящаю моим любимым
Эбигейл, Бетани и Шарлотте

Содержание

Предисловие xv
Благодарности xvi
Об этой книге xvii
Об авторе xxi

Часть 1 Основные сервисы Azure 1

1 *Перед началом работы 3*

1.1 Для кого предназначена эта книга? 3

1.2 Как пользоваться этой книгой 4

Основные главы 4 ▪ Попробуйте сейчас 5 ▪ Практические задания 5 ▪ Исходный код и вспомогательные материалы 5

1.3 Создание лабораторной среды 5

Создание бесплатной учетной записи Azure 5 ▪ Бонусное практическое упражнение: создайте бесплатную учетную запись GitHub 7

1.4 Немного помощи 8

1.5 Общие сведения о платформе Azure 8

Виртуализация в Azure 11 ▪ Инструменты управления 12

2 *Создание виртуальной машины 14*

2.1 Основы настройки виртуальной машины 15

Регионы и варианты доступности 15 ▪ Образы виртуальных машин 16 ▪ Размеры ВМ 17 ▪ Хранилище Azure 18 ▪ Виртуальные сети 19

2.2 Создание пары ключей SSH для аутентификации 20
2.3 Создание ВМ в браузере 22
2.4 Подключение к ВМ и установка веб-сервера 24
Подключение к ВМ с помощью SSH 24 ▪ Установка веб-сервера 26
2.5 Открытие доступа к ВМ для веб-трафика 27
Создание правила для разрешения веб-трафика 28 ▪ Обзор работы веб-сервера 28
2.6 Практическое упражнение: создание ВМ Windows 29
2.7 Очистка ресурсов 30
2.8 Хьюстон, у нас проблема 30

3 Azure Web Apps 33
3.1 Обзор сервиса Azure Web Apps и основные понятия 34
Поддержка языков и сред 34 ▪ Промежуточное хранение различных версий в слотах развертывания 35 ▪ Планы App Service 35
3.2 Создание веб-приложения 37
Создание простого веб-приложения 37 ▪ Развертывание образца сайта в формате HTML 39
3.3 Просмотр журналов диагностики 42
3.4 Практическое упражнение: создание и использование слота развертывания 44

4 Знакомство с сервисом хранилища Azure 47
4.1 Управляемые диски 47
Диски ОС 48 ▪ Временные диски и диски данных 49 ▪ Параметры кэширования дисков 50
4.2 Добавление дисков в ВМ 50
4.3 Хранилище Azure 52
Хранилище таблиц 53 ▪ Хранилище очередей 55 ▪ Доступность иизбыточность хранилища 56
4.4 Практическое упражнение: изучение сервиса хранилища Azure 57
Ориентация на ВМ 57 ▪ Ориентация на разработчика 57

5 Основные сведения о сервисе «Сети Azure» 58
5.1 Компоненты виртуальной сети 58
Виртуальные сети и подсети 59 ▪ Виртуальные сетевые адаптеры 61 ▪ Публичный IP-адрес и разрешение DNS 62
5.2 Защита и контроль трафика с помощью групп безопасности сети 64
Создание группы безопасности сети 64 ▪ Связывание группы безопасности сети с подсетью 66 ▪ Создание правил фильтрации для группы безопасности сети 67

5.3 Создание примера веб-приложения с защищенным трафиком 68

Создание сетевых подключений удаленного доступа 68 ▪ Создание ВМ 69 ▪ Использование агента SSH для подключения к виртуальным машинам 70

5.4 Практическое упражнение: установка и тестирование веб-сервера LAMP 72

Часть 2 Высокая доступность и масштабирование 73

6 Azure Resource Manager 75

6.1 Модель Azure Resource Manager 75

Проектирование на основе жизненного цикла приложений 76 ▪ Защита ресурсов и управление ими 78 ▪ Защита ресурсов с помощью блокировок 79 ▪ Контроль и группирование ресурсов с помощью тегов 80

6.2 Шаблоны Azure Resource Manager 81

Создание и использование шаблонов 82 ▪ Создание нескольких однотипных ресурсов 84 ▪ Средства для создания собственных шаблонов 85 ▪ Хранение и использование шаблонов 87

6.3 Практическое упражнение: развертывание ресурсов Azure на основе шаблона 87

7 Высокая доступность и избыточность 90

7.1 Необходимость избыточности 90

7.2 Избыточность инфраструктуры с зонами доступности 92

Создание сетевых ресурсов в зоне доступности 94 ▪ Создание виртуальных машин в зоне доступности 95

7.3 Избыточность для виртуальных машин с помощью групп доступности 96

Домены сбоя 97 ▪ Домены обновления 97 ▪ Распределение виртуальных машин в группе доступности 98 ▪ Просмотр распределения виртуальных машин по группам доступности 101

7.4 Практическое упражнение: развертывание виртуальных машин с высокой доступностью на основе шаблона 102

8 Приложения для балансировки нагрузки 106

8.1 Компоненты подсистемы балансировки нагрузки в Azure 106

Создание пула внешних IP-адресов 108 ▪ Создание и настройка зондов работоспособности 110 ▪ Определение распределения трафика с помощью правил подсистемы балансировки нагрузки 112 ▪ Маршрутизация прямого трафика с правилами преобразования сетевых адресов 114 ▪ Назначение групп виртуальных машин для внутренних пулов 116

8.2 Создание и настройка виртуальных машин с использованием подсистем балансировки нагрузки 119

8.3 Практическое упражнение: просмотр шаблонов существующих развертываний 122

9 Масштабируемые приложения 124

9.1 Зачем создавать масштабируемые и надежные приложения? 124

Масштабирование виртуальных машин по вертикали 125

- *Масштабирование веб-приложений по вертикали 128*
- *Масштабирование ресурсов по горизонтали 128*

9.2 Масштабируемые наборы виртуальных машин 129

Создание масштабируемого набора виртуальных машин 131

- *Создание правил автоматического масштабирования 133*

9.3 Масштабирование веб-приложения 136

9.4 Практическое упражнение: установка приложений в масштабируемом наборе или веб-приложении 139

Масштабируемые наборы виртуальных машин 139

- *Веб-приложения 140*

10 Глобальные базы данных с Cosmos DB 141

10.1 Что такое Cosmos DB? 141

Структурированные (SQL) базы данных 142 ▪ *Неструктурированные (NoSQL) базы данных 142* ▪ *Масштабирование баз данных 143*

- *Сведение ресурсов воедино с помощью Cosmos DB 144*

10.2 Создание учетной записи и базы данных Cosmos DB 145

Создание и заполнение базы данных Cosmos DB 145

- *Добавление глобальной избыточности в базу данных Cosmos DB 149*

10.3 Доступ к глобально распределенным данным 152

10.4 Практическое упражнение: развертывание веб-приложения, использующего Cosmos DB 156

11 Управление сетевым трафиком и маршрутизацией 158

11.1 Что такое Azure DNS? 158

11.2 Делегирование реального домена в сервис Azure DNS 160

11.3 Глобальная маршрутизация и разрешение адресов с помощью диспетчера трафика 162

Создание профиля диспетчера трафика 164

- *Глобальное распределение трафика в ближайший экземпляр 167*

11.4 Практическое упражнение: развертывание веб-приложений для просмотра диспетчера трафика в действии 174

12 Мониторинг и устранение неполадок 175

12.1 Диагностика загрузки виртуальных машин 175

12.2 Метрики производительности и оповещения о ней 178

Просмотр метрик производительности с помощью расширения системы диагностики виртуальной машины 178 ▪ Создание оповещений о состоянии производительности 181

12.3 Наблюдатель за сетями Azure 182

Проверка потоков IP-адресов 183 ▪ Просмотр действующих правил NSG 184 ▪ Захват сетевых пакетов 186

12.4 Практическое упражнение: создание оповещений о производительности 188

Часть 3 Безопасность по умолчанию 189

13 ***Резервное копирование, восстановление и репликация 191***

13.1 Сервис архивации Azure 191

Политики и хранение 193 ▪ Расписания резервного копирования 196 ▪ Восстановление виртуальной машины 198

13.2 Azure Site Recovery 201

13.3 Практическое упражнение: настройка виртуальной машины для использования сервиса Site Recovery 204

14 ***Шифрование данных 206***

14.1 Что такое шифрование данных? 206

14.2 Шифрование данных во время хранения 208

14.3 Шифрование сервиса хранилища 209

14.4 Шифрование виртуальной машины 211

Хранение ключей шифрования в Azure Key Vault 211 ▪ Шифрование виртуальной машины Azure 213

14.5 Практическое упражнение: шифрование виртуальной машины 214

15 ***Защита информации с помощью Azure Key Vault 216***

15.1 Защита информации в облаке 216

Программные хранилища и аппаратные модули безопасности 217 ▪ Создание хранилища ключей и секрета 219

15.2 Управляемые удостоверения для ресурсов Azure 221

15.3 Получение секрета из виртуальной машины с управляемым удостоверением 224

15.4 Создание и внедрение сертификатов 229

15.5 Практическое упражнение: настройка безопасного веб-сервера 232

16 Центр безопасности Azure и обновления 234

16.1 Центр безопасности Azure 234

16.2 Ограниченный по времени доступ 237

16.3 Управление обновлениями Azure 241

Объединенные сервисы управления Azure 243

▪ *Просмотр и применение обновлений 245*

16.4 Практическое упражнение: включение ограниченного по времени доступа и обновлений для виртуальной машины Windows 249

Часть 4 Самое интересное 251

17 Машинное обучение и искусственный интеллект 253

17.1 Обзор и взаимосвязь искусственного интеллекта и машинного обучения 254

Искусственный интеллект 254 ▪ *Машинное обучение 255* ▪ *Объединение искусственного интеллекта и машинного обучения 256* ▪ *Инструменты машинного обучения Azure для специалистов по аналитике данных 257*

17.2 Azure Cognitive Services 259

17.3 Создание интеллектуального бота, помогающего заказывать пиццу 260

Создание бота веб-приложения на платформе Azure 260 ▪ *Язык и концепция распознавания в LUIS 261* ▪ *Создание и запуск бота веб-приложения с LUIS 264*

17.4 Практическое упражнение: добавление каналов для связи с ботом 267

18 Автоматизация Azure 269

18.1 Что такое сервис автоматизации Azure? 269

Создание учетной записи сервиса автоматизации Azure 271

▪ *Ресурсы и модули Runbook сервиса автоматизации Azure 272*

18.2 Пример модуля Runbook сервиса автоматизации Azure 274

Запуск примера модуля Runbook и просмотр его выходных данных 276

18.3 Настройка требуемого состояния (DSC) PowerShell 278

Определение и использование PowerShell DSC и опрашивающего сервера сервиса автоматизации Azure 280

18.4 Практическое упражнение: использование DSC с Linux 282

19 Контейнеры Azure 284

19.1 Что такое контейнеры? 284

19.2 Подход к разработке и развертыванию приложений с использованием микросервисов 287

19.3 Экземпляры контейнеров Azure 289

19.4 Сервис Azure Kubernetes 293

Создание кластера с помощью сервиса Azure Kubernetes Services 294 ▪ *Запуск базового веб-сайта в Kubernetes 295*

19.5 Практическое упражнение: масштабирование развертывания Kubernetes 298

20 Azure и Интернет вещей 300

20.1 Что такое Интернет вещей? 300

20.2 Централизованное управление устройствами с помощью центра Интернета вещей Azure 303

20.3 Создание имитации устройства Raspberry Pi 306

20.4 Потоковая передача данных центра Интернета вещей Azure в веб-приложения Azure 309

20.5 Обзор компонентов Интернета вещей Azure 315

20.6 Практическое упражнение: изучение вариантов использования Интернета вещей 316

21 Бессерверные вычисления 317

21.1 Что такое бессерверные вычисления? 317

21.2 Платформы обмена сообщениями Azure 319

Сетка событий Azure 320 ▪ *Концентраторы событий и сервисная шина Azure 321* ▪ *Создание сервисной шины и ее интеграция с центром Интернета вещей 322*

21.3 Создание приложения логики Azure 325

21.4 Создание приложения-функции Azure для анализа данных устройства Интернета вещей 328

21.5 Не прекращайте обучение 332

Дополнительные учебные материалы 333 ▪ *Ресурсы GitHub 333*

Заключение 333

Алфавитный указатель 335

Предисловие

Второе издание книги *Научитесь работать с Azure за месяц* напоминает мне о том, что все быстро меняется и всегда нужно продолжать учиться. Канули в Лету времена, когда можно было за неделю пройти курс по операционной системе Windows Server и без проблем работать с ней в течение многих лет, практически ничего не меняя. Это не значит, что ИТ-мир стал страшнее, но для работы с облачными вычислениями необходимы открытый ум и готовность к постоянной адаптации.

Когда я начал работать с Azure, количество доступных сервисов было огромным. Я знал, что должен уделять внимание безопасности, производительности, избыточности и масштабированию, но не понимал, как применить десятилетний опыт крупномасштабного администрирования серверов в среде облачных вычислений. Со временем я начал знакомиться с различными сервисами Azure, которые предоставляют эти важные компоненты. Они редко работают изолированно, но я не знал, как наилучшим образом их интегрировать и как выбрать сервис для каждой задачи. Эта книга призвана объяснить мое прошлое самому себе и показать многим другим, идущим по схожему пути, как быстро освоить основные сервисы Azure и заставить их работать как единое целое.

В этой книге больше 350 страниц, но лишь поверхностно изложено то, что можно делать в Azure! Чтобы дать вам четкое представление о концепциях, необходимых для успешного создания решений в Azure, мне пришлось выбирать, какие темы рассмотреть. В книге не охватываются все 100 сервисов Azure и нет подробных сведений о включенных сервисах. Вместо этого внимание уделено самым важным вопросам, связанным с некоторыми основными сервисами, приведены примеры безопасного подключения ресурсов и представлены возможности, доступные в Azure.

Облачные вычисления постоянно меняются. Не существует 3- или 4-летних циклов выпуска и крупных развертываний обновлений. Я думаю, что сейчас самое время для создания решений и написания кода, так как всегда есть возможность узнать что-то новое и повысить свою квалификацию. Я надеюсь, что вы научитесь запускать прекрасные приложения в Azure и получите удовольствие от изучения всех доступных сервисов.

Благодарности

Эту книгу помогли опубликовать сотрудники издательства Manning Publications. Особая благодарность Майку Стивенсу (Mike Stephens) за то, что поделился своим видением для начала этого проекта. Спасибо моему издателю Марьяну Бэйсу (Marjan Bace) и всем сотрудникам редакционных и производственных групп. Я благодарю технических рецензентов во главе с Александром Драгосавлевичем (Aleksandar Dragosav-ljevic): Ариэля Гамино (Ariel Gamino), Чарльза Лама (Charles Lam), Эрнесто Карденаса Кангауалу (Ernesto Cardenas Cangahuala), Джорджа Онофрея (George Onofrei), Глена Томпсона (Glen Thompson), Хосе Апаблазу (Jose Apablaza), Юраджая Борзу (Juraj Borza), Майкла Лэнгдона (Michael Langdon), Майкла Уолла (Michael Wall), Питера Крейенхопа (Peter Kreyenhop), Рика Оллера (Rick Oller), Роба Рутча (Rob Ruetsch), Роберта Уолша (Robert Walsh) и Вишала Сингха (Vishal Singh). И наконец, спасибо Карстену Стрёбеку (Karsten Strøbaek) — техническому редактору и корректору книги.

За это второе издание я выражаю огромную благодарность Филу Эвансу (Phil Evans) и Давананду Бахаллу (Davanand Bahall) за их поддержку и свободу при обновлении этой книги. Над этим проектом я трудился после повседневной работы в Microsoft, но он затронул многих. Большое спасибо Дэвиду Толкову (David Tolkov) и Тиму Тибкену (Tim Teebken), которые дали мне возможность стать человеком, способным написать эту книгу. И, конечно, спасибо Жан-Полу Конноку (Jean-Paul Connock) — с момента выхода первого издания мы выиграли Кубок Стэнли! Это было прекрасно!

Благодарю Рика Клауса (Rick Claus) за поддержку потребности в хорошей технической документации по Azure, а также Марша Мэйси и Нила Петерсона за их личную поддержку и руководство при написании первой версии этой книги. Нам все еще нужно начать работу над этим школьным автобусом.

Об этой книге

Эта книга призвана дать вам прочную основу для достижения успеха в качестве ИТ-инженера или разработчика в Azure. Вы узнаете о решениях на основе модели «инфраструктура как сервис» (IaaS) и «платформа как сервис» (PaaS), а также о том, когда следует использовать каждый подход. В ходе работы над главами вы научитесь правильно планировать доступность и масштабирование, учитывать безопасность, а также рассматривать вопросы, связанные с затратами и производительностью. К концу книги вы сможете интегрировать технологии будущего — контейнеры и Kubernetes, искусственный интеллект и машинное обучение, а также Интернет вещей.

Что касается создания и запуска приложений и сервисов, Azure позволяет выбрать операционную систему, инструменты приложений и платформу, с которыми вам удобнее всего работать. В этой книге в основном рассматриваются технологии, разработанные не Microsoft, в частности Linux, Python и Node.js. В примерах команд используется интерфейс командной строки Azure, а не Azure PowerShell. Это были сознательные решения, чтобы показать, что использование Azure не означает необходимость применения Windows Server, IIS или ASP.NET.

Работая в облаке, вы часто используете разные платформы, поэтому вам приходится постоянно изучать новые темы. Это еще одна причина демонстрации технологий и платформ, разработанных не Microsoft. В этой книге вы ознакомитесь с некоторыми из этих новых тем, прежде чем столкнетесь с ними в реальной жизни. На протяжении всей книги вы будете узнавать концепции и действия, необходимые для интеграции сервисов Azure. Знание этих концепций и действий позволит вам переключать платформы и языки по своему желанию и применять одни и те же навыки.

Структура

Книга разделена на 4 части и содержит 21 главу.

- Часть 1 посвящена некоторым основным сервисам инфраструктуры и платформы Azure — виртуальным машинам, веб-приложениям, сервису хранилища и сетям.
- В части 2 описывается обеспечение высокой доступности и избыточности: рассматриваются шаблоны, группы и зоны доступности, подсистемы балансировки нагрузки, автомасштабирование, распределенные базы данных и маршрутизация трафика. К концу главы 12 вы должны хорошо понимать, как создавать высокопроизводительные распределенные приложения в Azure.
- В части 3 рассматриваются вопросы безопасности, в частности резервное копирование и восстановление, шифрование, управление цифровыми ключами и обновления. К тому времени, когда вы завершите чтение главы 16, вы уже будете на пути к созданию безопасных, стабильно работающих приложений в Azure.
- В части 4 рассматриваются новые технологии, такие как бессерверные вычисления и контейнерные приложения. В этих главах представлены области применения Azure, которые дают представление о том, как может выглядеть будущее рабочих приложений.

За исключением части 4, точно названной «Самое интересное», главы книги следует стараться читать по порядку. В последовательно расположенных главах не рассматривается какой-то один проект, но в каждой из них учитываются концепции и примеры практических упражнений, рассмотренные в предыдущих главах.

В главе 1 указано, как создать в Azure бесплатную пробную учетную запись, которой будет достаточно для выполнения практических упражнений во всех главах. В этой главе также немного подробнее рассказывается об Azure и о том, как найти дополнительные справочные сведения в ходе изучения материалов книги. Я упоминаю веб-страницу по адресу http://docs.microsoft.com/azure несколько раз в книге. Возможно, это выглядит немного предвзято, но я считаю, что это лучший ресурс для получения дополнительной документации и поддержки по любым интересующим вас вопросам использования Azure.

О примерах и исходном коде

В этой книге есть много примеров исходного кода как в виде нумерованных фрагментов кода, так и в виде строк с обычным текстом. В обоих случаях исходный код представляется **шрифтом фиксированной ширины, как этот**, чтобы отделить его от обычного текста.

Во многих случаях первоначальный формат исходного кода изменен: добавлены разрывы строк и скорректированы отступы, чтобы примеры помещались в доступное пространство страниц книги. В редких случаях, когда даже этого было недостаточно, в списки добавлены маркеры продолжения строки (➥). Кроме того, комментарии в исходном коде удалены из фрагментов кода, если код описан в тексте. Многие фрагменты кода сопровождаются аннотациями, в которых подчеркиваются важные концепции.

Исходный код этой книги, а также сопутствующие скрипты, шаблоны и вспомогательными ресурсами, доступны по адресу https://www.manning.com/books/learn-azure-in-a-month-of-lunches-second-edition и в репозитории GitHub этой книги (https://github.com/fouldsy/azure-mol-samples-2nd-ed).

Все практические упражнения можно выполнять на портале Azure и в Azure Cloud Shell — интерактивной оболочке на основе браузера для интерфейса командной строки Azure и Azure PowerShell. На компьютер не требуется устанавливать никакие инструменты, поэтому вы можете использовать любой компьютер и любую операционную систему, которая поддерживает современный веб-браузер.

На портале Azure часто происходят незначительные изменения. Одна из трудностей использования любого облачного сервиса заключается в том, что каждый день он может выглядеть по-новому.

Во втором издании я попытался свести к минимуму количество снимков портала, но не волнуйтесь, если то, что вы увидите, будет немного отличаться от показанного в книге. Обязательные параметры обычно одинаковы — просто формат может быть другим. Если на портале есть новые параметры, которые не указаны в практическом упражнении, обычно можно смело принимать значения по умолчанию.

При работе не в Azure Cloud Shell будьте внимательны с примерами команд. Оболочки на основе Windows, в частности PowerShell и CMD, обрабатывают разрывы и продолжения строк не так, как оболочки на основе *nix, например Azure Cloud Shell. Во многих примерах команд используются несколько строк. Команды показываются с символом обратной косой черты (\), означающем, что команда продолжается на следующей строке:

```
az resource group create \
--name azuremol \
--location eastus
```

Вам эти символы обратной косой черты вводить не нужно, но это может сделать длинные команды удобочитаемыми на экране. Если вы будете работать локально на компьютере с оболочкой Windows, можете вместо обратной косой черты использовать обратную кавычку (`). Например, в оболочке PowerShell или CMD с установленным ПО Python для Windows измените предыдущую команду следующим образом:

```
az resource group create `
--name azuremol `
--location eastus
```

Поначалу это может сбивать с толку, но я придерживаюсь данного соглашения в книге, потому что в официальной документации https://docs.microsoft.com/azure используется такой формат. Команды Azure CLI, которые в основном применяются в этой книге, предполагают использование оболочки на основе *nix, поэтому в них вводятся символы обратной косой черты. Команды Azure PowerShell предполагают использование оболочки на основе Windows, поэтому в них вводятся обратные кавычки. Вы быстро привыкните к этому и сможете легко переключаться между двумя оболочками. Если вы новичок в работе на разных платформах, это может оказаться забавным «подводным камнем»!

Если вы используете Windows 10 и хотите подробнее изучить Azure CLI и системы на основе *nix в целом, я рекомендую вам ознакомиться с подсистемой Windows для Linux (WSL) здесь: https://docs.microsoft.com/windows/wsl. WSL и последние улучшения в WSL2 предоставляют возможности работы с ядром Linux при использовании Windows. Не пытайтесь сразу окунуться во все аспекты этой темы. Просто знайте, что вы можете выполнять собственные команды и приложения Linux, не беспокоясь о различных разрывах строк или определениях переменных. Что еще лучше, оболочка PowerShell доступна для платформы .NET Core, которая также работает в Linux. Вы можете запустить PowerShell в Linux, работая в Windows.

Дискуссионный форум liveBook

Купив книгу *Научитесь работать с Azure за месяц*, вы получаете бесплатный доступ к веб-форуму, организованному издательством Manning Publications, где можете оставлять комментарии о книге, задавать технические вопросы и получать помощь от автора и других пользователей. Форум доступенпо адресу https://livebook.manning.com/book/learn-azure-in-a-month-of-lunches-second-edition/discussion. Узнать подробнее о форумах Manning и правилах участия в них можно также по адресу https://livebook.manning.com/discussion.

Издательство Manning берет на себя обязательство предоставить читателям место для проведения содержательного диалога между собой, а также с автором. Это не обязательство обеспечить какой-либо конкретный объем участия со стороны автора, чей вклад в форум остается добровольным (и неоплачиваемым). Мы предлагаем вам попробовать задать ему несколько сложных вопросов, чтобы поддержать его интерес! Форум и архивы предыдущих обсуждений будут доступны на веб-сайте издателя до тех пор, пока книга находится в печати.

Об авторе

ИЭН ФОУЛДС (IAIN FOULDS) — ведущий разработчик контента в корпорации Microsoft, который в настоящее время пишет техническую документацию по Azure Active Directory. Ранее Иэн был ведущим инженером-наладчиком в Microsoft по технологиям виртуализации — Azure, Hyper-V и System Center Virtual Machine Manager. Иэн, имеющий 15-летний опыт работы в области ИТ (большей частью в сфере эксплуатации и сервисов), рано внедрил виртуализацию с использованием решений VMware и в течение многих лет помогал создавать решения для облачных вычислений и учить этому других.

Иэн родом из Англии, в США приехал 10 лет назад. В настоящее время проживает недалеко от Сиэтла с женой и 2 маленькими детьми, которым посвящена эта книга. Он является поклонником футбола (к сожалению, называемого «соккером» там, где он живет). Ему также нравится хоккей с шайбой и почти все виды автогонок. Помимо работы с компьютерными системами, Иэн любит посещать зрелищные мероприятия, интересуется классическими автомобилями, авиационной фотографией и играет на гитаре. Он также большой любитель моделей поездов, регулярно посещает соответствующие выставки и мероприятия по всему тихоокеанскому северо-западу и работает там на общественных началах.

Часть 1

Основные сервисы Azure

Чтобы создать очередное успешное приложение, вы должны хорошо понимать базовые ресурсы Azure. Возможно, сети и хранилища не так интересны, но именно они — основа большинства выполняемых в Azure задач. Прежде чем работать с избыточными многоэкземплярными виртуальными машинами или веб-приложениями Azure, полезно изучить доступные параметры и задачи управления для одного экземпляра. Это позволяет узнать о различиях и сходствах в подходах IaaS с использованием ВМ и PaaS с использованием веб-приложений. В главах 1–5 мы рассмотрим виртуальные машины и веб-приложения, а также основные функции хранения и виртуальной сети.

1 Перед началом работы

Azure — одно из крупнейших публичных облаков, которое предоставляет виртуальные машины (ВМ), контейнеры, бессерверные вычисления, машинное обучение и другие сервисы. Мы не будем рассматривать больше 100 сервисов Azure в этой книге, но вы узнаете об основных и самых необходимых сервисах и функциях, позволяющих создавать и запускать решения в Azure. Мы рассмотрим типовой пример создания и запуска веб-приложения. Вы научитесь упрощать работу с помощью части основной инфраструктуры и сервисов платформы.

С Azure вам не нужен хрустальный шар, чтобы предсказывать количество серверов или объем хранилищ на ближайшие 3 года. Больше не нужно ждать, пока утвердят бюджет, поставят новое оборудование, установят и настроят его. Вам не нужно переживать об установленных версиях ПО или библиотек, пока вы пишете код.

Вместо всего этого просто нажмите кнопку и создайте все необходимые ресурсы. Вы платите только за время работы этих ресурсов либо за используемый объем хранилища или пропускную способность сети. Неиспользуемые ресурсы можно отключить или удалить. А если вам понадобится срочно увеличить вычислительную мощность в 10 раз, нажмите кнопку, подождите пару минут — и готово. И все это делает кто-то другой, позволяя вам сосредоточиться на приложениях и клиентах.

1.1 Для кого предназначена эта книга?

ИТ-индустрия все еще находится на переходном этапе, если говорить о названиях должностей. Вы можете называть себя ИТ-специалистом, разработчиком ПО, системным администратором или инженером DevOps. Но если вы хотите получить основные навыки для создания и запуска безопасных приложений высокой

доступности в облаке, эта книга для вас. Скорее всего, вы занимаетесь ИТ-операциями или разработкой. Правда в том, что вы часто меняете специализацию, особенно работая с облачными вычислениями. Независимо от того, занимаетесь ли вы разработкой или операционной деятельностью, вам важно понимать основную инфраструктуру и сервисы платформы, чтобы создавать и выполнять приложения, которые верой и правдой послужат вашим клиентам.

Во втором издании этой книги объясняется ряд основных понятий Azure, а также описываются навыки, необходимые для принятия обоснованных решений. Для работы с этой книгой вам понадобится опыт работы с ВМ и базовые знания в сфере сетей и хранилищ. Также желательно уметь создавать простые сайты и понимать, что такое SSL-сертификат и база данных. После того как мы изучим основные процессы, мы кратко рассмотрим новые и развивающиеся технологии. Вам всегда нужно идти в ногу с прогрессом, чтобы эффективно ориентироваться в рабочей среде. Поэтому вы узнаете о контейнерах, Интернете вещей, машинном обучении, искусственном интеллекте и бессерверных вычислениях. Все, кто называет себя «разработчиком» и «ИТ-специалистом», откроют совершенно новые области для изучения!

1.2 Как пользоваться этой книгой

Я люблю бутерброды, поэтому обед — лучшее время для изучения перспективных технологий. Возможно, совы выделят немного свободного времени вечером, а, может быть, вы жаворонок (как вам это удается?!) и с удовольствием разберете главу за завтраком. Правильного времени для обучения не существует, но всего за 45 минут в день вы успеете прочитать одну главу и выполнить упражнения. Каждая глава рассказывает о чем-то новом, поэтому дайте себе время, чтобы усвоить ежедневный урок.

1.2.1 Основные главы

Книга разделена на 4 части — по количеству недель в месяце.

- Часть 1 (главы 1–5) охватывает основные ресурсы Azure. Именно эти главы дают четкое понимание. Затем можете выбрать наиболее интересные для вас главы.
- Часть 2 (главы 6–12) посвящена доступности и масштабированию. Вы узнаете, как автоматически масштабировать ресурсы, распределять трафик и обрабатывать события обслуживания без простоев. Здесь описано, как запускать высокодоступные приложения на глобальном уровне.
- Часть 3 (главы 13–16) предназначена для энтузиастов безопасности. Здесь описано, как шифровать ВМ, хранить SSL-сертификаты в безопасном хранилище, создавать и восстанавливать резервные копии данных.
- Часть 4 (главы 17–21) охватывает наиболее интересные вопросы о пользе Azure для вас и ваших клиентов. Мы рассмотрим автоматизацию, контейнеры, Интернет вещей и бессерверные вычисления. Выбирайте то, что вам по душе, и радуйтесь жизни!

1.2.2 Попробуйте сейчас

Хотите просто читать или, засучив рукава, поиграть с Azure? В книге есть небольшие задания, позволяющие быстро опробовать что-то новое. Если у вас есть время, попробуйте их выполнить. Большинство практических заданий приведены в упражнениях в конце каждой главы, но вы достигнете большего, если иногда будете отрываться от чтения и применять новые понятия. Для некоторых из упражнений доступны пошаговые инструкции. Во время выполнения других придется подумать немного усерднее и научиться создавать решения самостоятельно, как в реальном мире.

1.2.3 Практические задания

В конце каждой главы есть практические задания. В некоторых главах, например в этой, вы найдете упражнения в середине главы. Выполняя их, выпоймете, как все элементы Azure взаимодействуют между собой, а также натренируете память. Садитесь за клавиатуру и создайте что-нибудь потрясающее!

1.2.4 Исходный код и дополнительные материалы

Исходный код этой книги, а также сопутствующие скрипты, шаблоны и вспомогательными ресурсами, доступны по адресу https://www.manning.com/books/learn-azure-in-a-month-of-lunches-second-edition и в репозитории GitHub этой книги (https://github.com/fouldsy/azure-mol-samples-2nd-ed). Кроме того, вы можете посетить форум, посвященный книге, по адресу https://livebook.manning.com/book/learn-azure-in-a-month-of-lunches-second-edition/discussion.

1.3 Создание лабораторной среды

Книга не перегружена понятиями и архитектурой, зато в ней описано много практических нюансов по работе с платформой Azure. Для этого вам понадобится учетная запись Azure.

1.3.1 Создание бесплатной учетной записи Azure

Azure предлагает бесплатную пробную учетную запись на 30 дней и до 200 долларов США на счет. Бонусных средств должно хватить, чтобы пройти все главы, выполнить упражнения, немного поэкспериментировать и получить массу удовольствия. Многие сервисы и функции Azure остаются бесплатными даже по истечении пробного периода.

Попробуйте сейчас

Чтобы создать бесплатную учетную запись Azure, выполните действия, указанные в этом разделе:

1. Откройте в браузере страницу https://azure.microsoft.com/free и выберите параметр, чтобы начать работу с бесплатной учетной записью Azure.
2. По запросу выполните вход в учетную запись Microsoft. Если у вас нет учетной записи Microsoft, перейдите по ссылке «Создать учетную запись Microsoft».

3. Войдя в учетную запись Microsoft, следуйте подсказкам, чтобы создать бесплатную учетную запись Azure:
   - Введите свои личные данные в соответствии с запросом.
   - Для защиты от мошенничества и злоупотреблений укажите свой номер телефона. Это позволит подтвердить вашу личность с помощью SMS или звонка.
   - Для подтверждения личности также требуется номер кредитной карты, но здесь нет подвоха. Система начнет выставлять счета только через 30 дней или когда ваш бонус на сумму 200 долларов США будет исчерпан. В конце пробного периода нет автоматического перехода на платную подписку. Вы можете увидеть списание на небольшую сумму (1 доллар США или эквивалент в местной валюте), которая будет возвращена через несколько дней.
4. Просмотрите и примите соглашение о подписке на Azure и политику конфиденциальности, а затем выберите «Зарегистрироваться». Через несколько минут ваша подписка на Azure будет готова.
5. После регистрации и загрузки портала Azure просмотрите краткий обзор системы.

Ваша панель (домашняя страница портала) сейчас выглядит пустой. Но уже в главе 2 вы создадите свою первую ВМ, и панель приобретет вид, как на рисунке 1.1!

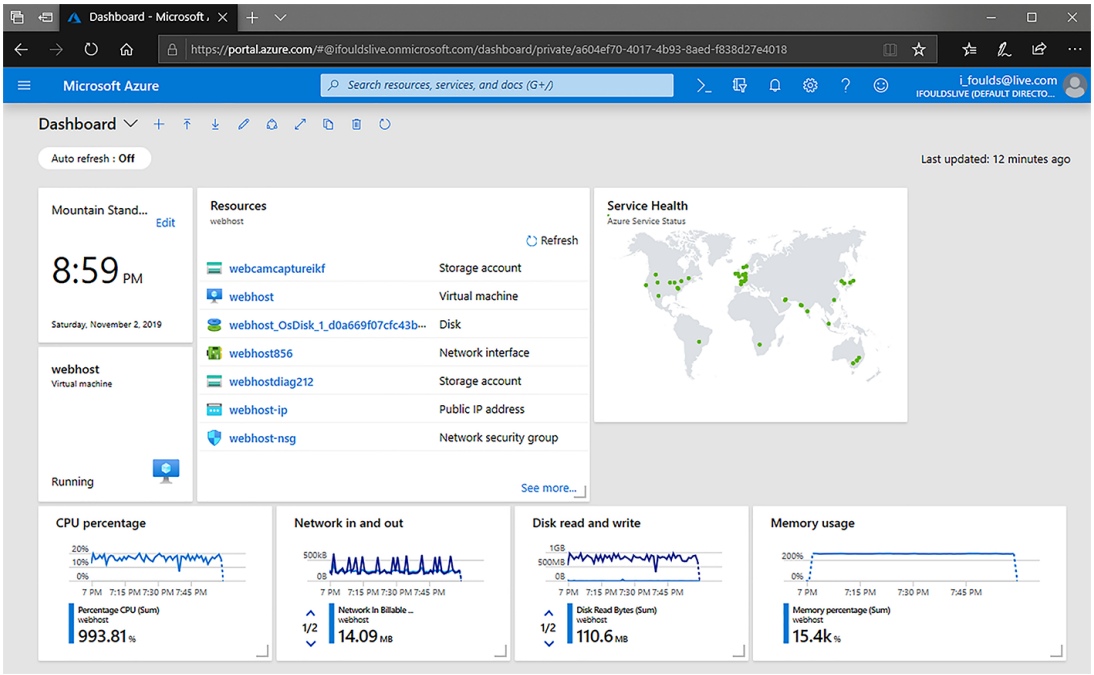


Рис. 1.1. Портал Azure готов к созданию ваших приложений и решений.

Действительно бесплатно?

В Azure есть Marketplace с сотнями готовых образов (основа ВМ) и решений для развертывания. В этой книге мы используем некоторые предложения Marketplace, потому что это отличный способ быстро развернуть целый набор приложений.

Не все предложения на Azure Marketplace бесплатны. Некоторые издатели включают стоимость лицензий или поддержки в развертываемое решение. Например, при развертывании ВМ Red Hat поддержка и лицензия оплачиваются дополнительно. Пробный бонус не покрывает эти расходы, он оплачивает использование только базовой ВМ.

В упражнениях этой книги используются только ресурсы бесплатной пробной версии. Если же вы экспериментируете с другими превосходными предложениями Marketplace в Azure, будьте внимательны, создавая что-либо. Если решение включает дополнительные платежи, оно должно четко сообщить об этом перед развертыванием!

1.3.2 *Бонусное практическое упражнение: создайте бесплатную учетную запись GitHub*

GitHub — это бесплатный веб-сервис, с помощью которого многие организации и специалисты управляют кодом, шаблонами и документацией. В Azure есть сотни бесплатных шаблонов и примеров сценариев, которые можно использовать и улучшать. Одно из преимуществ сообщества разработчиков ПО с открытым исходным кодом — общий доступ и взаимопомощь.

Некоторые упражнения в этой книге используют ресурсы из GitHub. Учетная запись GitHub необязательна для выполнения упражнений, но без нее вы не сможете сохранить изменения и создать свою коллекцию шаблонов и сценариев. Создавать учетную запись GitHub необязательно, но мы настоятельно рекомендуем ее как часть вашей лабораторной среды.

1. Откройте в браузере страницу https://github.com.
2. Чтобы создать бесплатную учетную запись GitHub, введите имя пользователя, адрес электронной почты и пароль. Вы получите сообщение о проверке от GitHub.
3. Перейдите по ссылке в сообщении электронной почты, чтобы активировать свою учетную запись.
4. Ознакомьтесь с рядом репозиториев Azure с образцами ресурсов:
   - Шаблоны быстрого запуска Azure — https://github.com/Azure/azure-quickstart-templates
   - Azure CLI — https://github.com/Azure/azure-cli
   - Служебные программы Azure DevOps — https://github.com/Azure/azure-devops-utils
   - Книга *Научитесь работать с Azure за месяц* — https://github.com/fouldsy/azure-mol-samples-2nd-ed

1.4 Немного помощи

Эта книга не охватывает все предложения Azure. Даже попытайся я вместить их в книгу, готов поспорить, ко времени ее выхода в Azure появилось бы что-то еще! Облачные вычисления быстро развиваются, постоянно появляются новые сервисы и функции. Возможно, я излишне настойчив, но сайт https://docs.microsoft.com/azure — это лучшее место, чтобы начать изучение дополнительных сервисов Azure. Для всех сервисов Azure есть краткие руководства, учебники, примеры кода, рекомендации разработчиков и руководства по созданию архитектуры. Кроме того, вы сможете обратиться за бесплатной или платной поддержкой.

1.5 Общие сведения о платформе Azure

Прежде чем продолжить чтение, давайте вернемся на шаг назад и разберемся, что такое Azure и доступные на этой платформе сервисы. Как я уже говорил, Azure — это поставщик облачных вычислений на глобальном уровне. На момент написания книги в мире было сформировано 54 региона Azure. В каждом регионе есть как минимум один центр обработки данных. Для сравнения два других главных облачных поставщика работают в 23-ти (AWS) и 20-ти (Google Cloud) регионах.

Поставщики облачных вычислений не только обрабатывают ресурсы. Azure предоставляет больше 100 сервисов, сгруппированных по семействам, например вычислительная инфраструктура, Интернет и мобильные устройства, контейнеры и удостоверения. С помощью всех этих сервисов Azure предоставляет множество моделей обслуживания. Кусочек пиццы на обед поможет нам понять, что это значит: см. рисунок 1.2.

Рис. 1.2. Модель «Пицца как сервис». По сравнению с домашней пиццей, где вы отвечаете за все, в ресторане, куда вы просто приходите, совершенно другие требования к управлению и степень ответственности.

В модели «Пицца как сервис» есть четыре варианта на выбор. По мере продвижения по моделям вы все меньше заботитесь о процессе поедания пиццы:

- *Домашняя* — вы замешиваете тесто; добавляете соус, начинку и сыр; выпекаете пиццу в духовке; достаете напитки; садитесь за обеденный стол и едите.
- *Полуфабрикат* — вы покупаете практически готовую пиццу. Нужно просто испечь ее в духовке, достать напитки, сесть за обеденный стол и поесть.
- *Доставка на дом* — вы заказываете пиццу с доставкой к вам домой. Просто достаньте напитки и сядьте за обеденный стол, чтобы поесть.
- *Ресторан* — вы хотите съесть пиццу вне дома с минимальными усилиями!

А теперь, когда вы проголодались, давайте взглянем на традиционную модель, включающую несколько вычислительных ресурсов: см. рисунок 1.3. Это уже ближе к тому, что вы видите в Azure.

Рис. 1.3. Модель сервисов облачных вычислений

По мере прохождения по моделям вы управляете меньшим количеством базовых ресурсов и можете уделять больше времени и энергии своим клиентам:

- *Локально* — вы настраиваете и контролируете весь центр обработки данных, например сетевые кабели, хранилища и серверы. Вы отвечаете за все элементы среды приложения, поддержку и избыточность. Такой подход дает максимальный контроль, но требует больше усилий по управлению.

- *Инфраструктура как сервис (IaaS)* — вы покупаете основные вычислительные ресурсы у поставщика, который управляет базовой инфраструктурой. Вы создаете и контролируете ВМ, данные и приложения. Поставщик облачных услуг отвечает за физическую инфраструктуру, управление хостами и устойчивость. У вас по-прежнему может быть группа специалистов по инфраструктуре, помогающая поддерживать и развернуть ВМ, но они не будут тратить время и деньги на управление физическим оборудованием.

  Этот подход хорош, если вы только начинаете перемещать приложения из собственной локальной среды. Управление и операции обычно схожи с локальными, поэтому IaaS обеспечивает естественный и комфортный переход в облако для компании, ИТ-сервисов и владельцев приложений.
- *Платформа как сервис (PaaS)* — вы покупаете базовый стек платформы у поставщика, управляющего ОС и исправлениями, и переносите туда свои приложения и данные. Вы не беспокоитесь о ВМ или виртуальной сети, а ваша операционная группа может сосредоточится на надежности и производительности приложений.

  Такой подход часто свидетельствует о начале организации ИТ-сервисов и о том, что компания увереннее запускает свои приложения в облаке. В данном случае, центр внимания — приложения и клиенты. Вы гораздо меньше переживаете об инфраструктуре, необходимой для запуска этих приложений.
- *Программное обеспечение как сервис (SaaS)* — вам просто нужен доступ к программному обеспечению, а поставщик обеспечивает все остальное. Разработчики могут создавать приложения на существующей платформе, чтобы предоставлять настройки или уникальные функции без большой базы кода.

  Сначала такой подход вызывает сомнения, но вы, вероятно, уже знаете и используете такие успешные предложения SaaS, как Salesforce, Office 365, пакеты программ Google Mail или Google Docs. Вы пользуетесь электронной почтой, создаете документы или презентации, управляете контактной информацией клиентов и сведениями о продажах. Здесь акцент делается на создаваемом и управляемом вами контенте, а не на выполнении приложения.

Многое из того, что вы создаете в Azure, относится к IaaS и PaaS. Основные варианты использования: виртуальные машины и виртуальные сети (IaaS) или веб-приложения, функции и сервисы Cosmos DB (PaaS) в Azure. Для разработчиков наиболее интересны решения PaaS, поскольку Microsoft управляет элементами инфраструктуры, освобождая вам время для написания кода. ИТ-специалистам больше подойдут решения IaaS для создания и контроля инфраструктуры Azure.

Всегда учитесь новому

Помните: даже когда компания переходит от IaaS- к PaaS-модели, ИТ-специалисты остаются при деле! Важно понимать, что происходит ниже уровня PaaS при разработке или устранении неполадок решения. Если вы ИТ-специалист, не пропускайте главы о решениях PaaS в Azure. Ваша компания и клиенты получат массу преимуществ, если вы поймете суть перехода к этой модели развертывания.

1.5.1 Виртуализация в Azure

Виртуализация — это настоящая магия, краеугольный камень в фундаменте Azure. Модели IaaS, PaaS и SaaS используют виртуализацию, чтобы обеспечить работу сервисов. Понятие виртуализации не ново, ее история началась в далекие дни мейнфреймов в 1960-х. В середине 2000-х виртуализация серверов в ЦОД начала набирать обороты, и теперь подавляющее большинство рабочих нагрузок виртуализируются, а не развертываются на серверах без операционной системы.

Виртуализации посвящены целые книги, но вкратце она логически делит физические ресурсы сервера на виртуальные с безопасным доступом для отдельных рабочих нагрузок. ВМ — один из самых распространенных ресурсов в облачных вычислениях. ВМ содержит виртуальный ЦП (vCPU), виртуальную память (vRAM), виртуальное хранилище (vDisk) и виртуальное сетевое соединение (vNIC), как показано на рисунке 1.4.

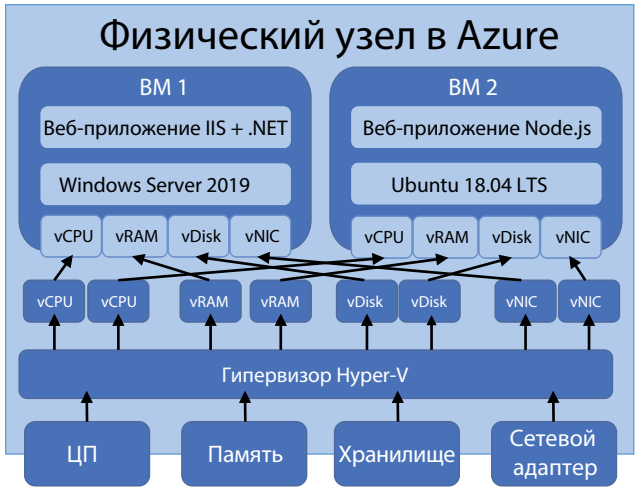


Рис. 1.4. Виртуализация в действии на физическом хосте в Azure

В дополнение к физическим серверам, есть виртуальные хранилища и сети. Это позволяет платформе Azure быстро определить все, что необходимо для ПО. Не требуются физическое взаимодействие или настройка устройств вручную. Не нужно ждать IP-адрес от другой группы разработчиков, открывать сетевой порт или добавлять хранилище.

По сути, Azure выполняется в среде, похожей на Windows. Измененная версия гипервизора Hyper-V обеспечивает работу вычислительных серверов. Hyper-V — это гипервизор типа 1 (без операционной системы), доступный в Windows Server уже больше 10 лет. Не переживайте, вы сможете выполнять Linux как полностью совместимую, первоклассную рабочую нагрузку! Корпорация Microsoft внесла огромный вклад в развитие сообщества и ядра Linux. Некоторые из основных программно-определяемых сетей в Azure управляются настраиваемым решением на основе Debian Linux — ПО для открытых сетей в облаке (SONiC), исходный код которого был открыт по инициативе Microsoft. Виртуальный обзор центров обработки данных Microsoft доступен по адресу https://azure.microsoft.com/global-infrastructure.

1.5.2 *Инструменты управления*

Как использовать столько сервисов Azure? Как хотите! Если предпочитаете работать в веб-браузере, есть великолепный интернет-портал. Дружите с PowerShell? Как вы уже догадались, имеется модуль Azure PowerShell. Еще один полезный инструмент — кросс-платформенный интерфейс командной строки (CLI). Он отлично подойдет для macOS или Linux. Разработчики могут взаимодействовать с Azure через интерфейсы REST API на таких популярных языках, как .NET, Python и Node.js.

Портал Azure

Портал Azure работает в любом современном веб-браузере, что очень удобно, поскольку не нужно устанавливать ПО на компьютер. Кроме того, портал — это наглядный и быстрый источник знаний. Здесь вы можете научиться создавать ресурсы и управлять ими.

В Azure постоянно появляются новые функции и сервисы, поэтому текущий вид портала может немного отличаться от снимков экрана, приведенных в этой книге, электронной документации или в блогах. Вы можете встретить другие названия кнопок или новые варианты выбора, но основные операции остаются неизменными. Добро пожаловать в дивный новый мир облачных вычислений!

Azure Cloud Shell

Если вы хотите набирать команды на клавиатуре, на портале есть Azure Cloud Shell. См. рисунок 1.5. Это интерактивная веб-консоль, предоставляющая оболочку Bash, Azure CLI и ряд предустановленных инструментов для разработки приложений, например Git и Maven. Кроме того, имеется облачная версия оболочки PowerShell, которая, как следует из названия, обеспечивает доступ к последним версиям командлетов Azure PowerShell.

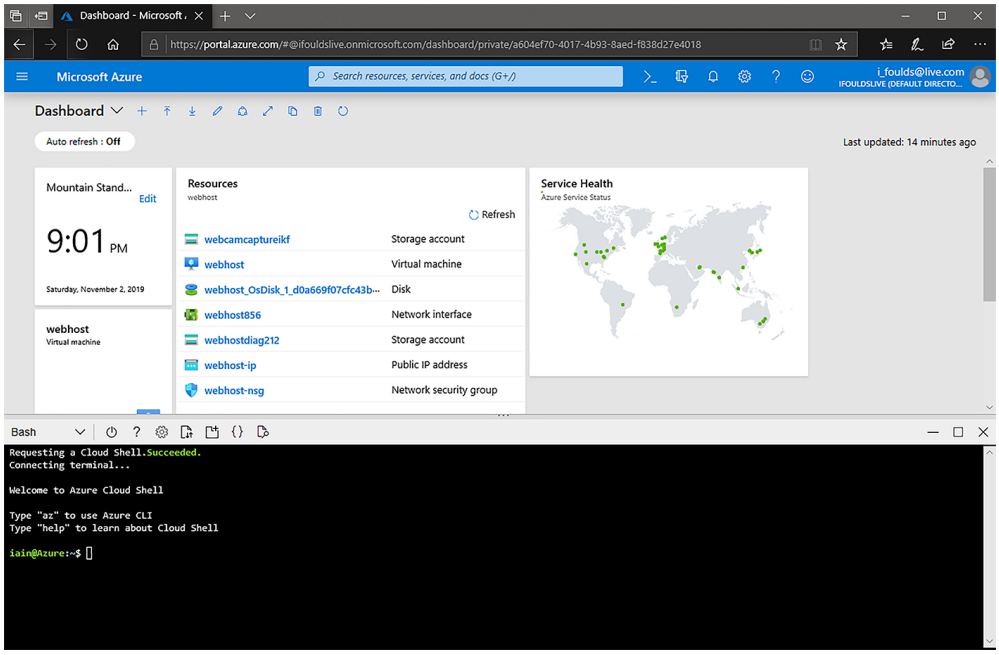


Рис. 1.5. Azure Cloud Shell на веб-портале

Программа Azure Cloud Shell не требует установки каких-либо инструментов и доступна в браузере на любом компьютере по адресу https://shell.azure.com. Такие редакторы, как Visual Studio Code (https://code.visualstudio.com) предоставляют доступ к Cloud Shell через приложение. Не забываем и о приложении Azure для iOS и Android, позволяющем использовать Azure Cloud Shell непосредственно на вашем смартфоне.

С помощью Azure Cloud Shell вы получаете доступ к самым последним версиям инструментов CLI или PowerShell. Постоянное хранилище позволяет создавать и хранить скрипты, шаблоны и файлы конфигурации.

Локальные инструменты Azure CLI и PowerShell

Несмотря на все преимущества Azure Cloud Shell, часто требуется доступ к локальным файловым системам и инструментам. Вы можете установить Azure CLI или Azure PowerShell в физической системе и работать с локальными ресурсами и ресурсами Azure.

В этой книге мы, как правило, используем Azure CLI (строго говоря, Azure CLI 2.0). Такой выбор может показаться странным, ведь мы могли выбрать собственную программу Microsoft — PowerShell. Но преимущество в том, что примеры и упражнения будут работать и в Azure Cloud Shell, и локально на вашем компьютере независимо от используемой ОС. Хотя это и не требуется для настройки вашей лабораторной среды, мы все-таки приведем руководства по установке инструментов управления Azure на компьютер.

- *Начните работу с Azure PowerShell* — https://docs.microsoft.com/powershell/azure/get-started-azureps
- *Установите Azure CLI* — https://docs.microsoft.com/cli/azure/install-azure-cli

2 Создание виртуальной машины

Вы готовы узнать, как быстро настроить веб-сервер в Azure? В этой главе мы сразу переходим к одному из наиболее частых вопросов о ВМ: создание простого веб-сервера. Эта рабочая нагрузка — отличный пример основных компонентов модели «Инфраструктура как сервис» (IaaS) в Azure.

Предположим, вы работаете в пиццерии и хотите принимать онлайн-заказы на доставку или самовывоз пиццы. Для работы в Интернете вам нужен веб-сайт. В первых главах книги мы исследуем различные функции и сервисы в Azure, позволяющие создавать и запускать веб-приложения IaaS и PaaS. Вы сможете обоснованно выбирать, как обеспечить работу веб-сайта: создавать и выполнять ВМ или воспользоваться PaaS. Но первый шаг — это создание веб-сервера.

В этой главе вы создадите ВМ Ubuntu Linux и установите простой веб-сервер. Не беспокойтесь об использовании Linux — в конце главы есть практическое задание по созданию ВМ Windows! Ubuntu — это основная платформа веб-серверов и отличный

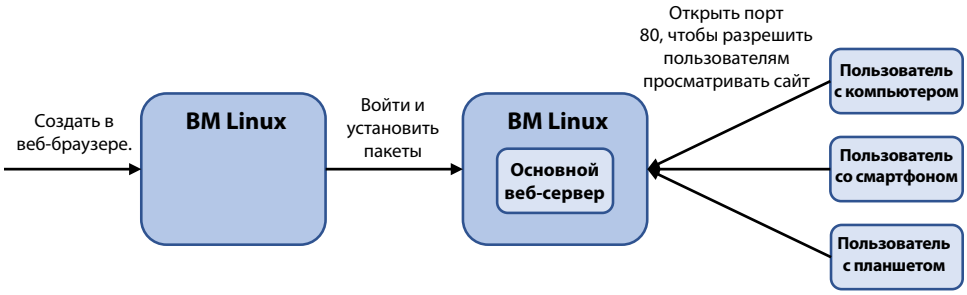


Рис. 2.1. В этой главе вы создадите простую ВМ, выполните вход в систему, установите веб-сервер, а затем откроете сетевой порт, чтобы разрешить клиентам переходить к образцу веб-сайта.

способ узнать о проверке подлинности по открытому ключу SSH. Затем вы увидите, как открыть сетевой порт, чтобы клиенты могли посещать ваш веб-сайт в Интернете. Общий обзор основной среды показан на рисунке 2.1.

2.1 Основы настройки виртуальной машины

Виртуальные машины — одни из самых распространенных стандартных блоков, которые будут использоваться при запуске приложений в облаке. Почему? Как правило, потому что они всем знакомы. Большинство ИТ-отделов выполняют множество рабочих нагрузок, используя Hyper-V или VMware в локальной среде, поэтому вы, вероятно, уже получили определенный опыт создания и запуска виртуальных машин. Часто первыми шагами организаций в Azure оказываются операции с виртуальными машинами, так как для запуск рабочих нагрузок IaaS не требует серьезных изменений образа мышления, которые необходимы для PaaS.

Есть решения для миграции виртуальных машин из локальной среды, такой как Hyper-V или VMware, в Azure. Но прежде чем слишком углубляться во все возможности Azure (некоторые из которых мы рассмотрим в следующих главах), рассмотрим основы. Следующие несколько страниц могут показаться вам знакомыми рекомендациями и вариантами, которые доступны для локальных виртуальных машин. Если это так, отлично! Если для вас это в новинку, не волнуйтесь! Большая часть задач управления абстрагируется в Azure, а такие ресурсы, как виртуальные сети, обычно создаются и настраиваются один раз, а затем работают без изменений. В следующих главах мы подробнее изучим каждую область, поэтому сделайте глубокий вдох и делайте шаг за шагом.

2.1.1 Регионы и варианты доступности

Платформа Azure разделена на регионы по всему миру, в каждом из которых есть один или несколько центров обработки данных. Они предоставляют базовые вычислительные ресурсы, хранилище и сетевые ресурсы для выполнения приложений и рабочих нагрузок. Azure работает в 50 регионах, и этот список растет каждые несколько месяцев. Идея заключается в том, чтобы вы могли развертывать приложения рядом с вашими сотрудниками или клиентами. Это уменьшает задержку и улучшает взаимодействие с конечными пользователями.

В определенном регионе Azure могут быть доступны не все сервисы Azure. Платформа предлагает сотни сервисов, но самый базовый набор, как правило, доступен везде. Однако новые или специализированные сервисы обычно развертываются с течением времени. При планировании запуска приложений в Azure проверьте доступность продуктов по регионам на странице https://azure.microsoft.com/global-infrastructure/services.

В главе 8 мы рассмотрим некоторые параметры высокой доступности, такие как группы доступности и зоны доступности. Эти параметры избыточности позволяют Azure распределять несколько экземпляров виртуальных машин или приложений в пределах одного центра обработки данных или по всему региону. Так вы можете сами определить допустимую толерантность к обслуживанию из-за обновлений или к сбоям оборудования. В первых главах этой книги вы чаще всего будете создавать только 1 или 2 виртуальные машины, поэтому не стоит беспокоиться об этих вариантах доступности раньше времени.

2.1.2 *Образы виртуальных машин*

Чтобы создать виртуальную машину, нужна отправная точка. Как правило, она сводится к выбору операционной системы: Windows или Linux. Далее необходимо выбрать, какую версию Windows (например, Windows Server 2016 или 2019) или какой дистрибутив Linux (например, Ubuntu, Red Hat Enterprise Linux или SUSE) вы будете использовать.

Образ — это предварительно настроенный пакет ОС с примененными базовыми параметрами конфигурации, который служит отправной точкой. В Azure Marketplace есть сотни готовых образов для использования при создании виртуальных машин. Часто можно применять имеющиеся лицензии Windows в зависимости от текущей модели лицензирования. Или вы можете выбрать дополнительную поддержку Canonical для запуска Ubuntu Linux или получения обновлений от Red Hat, например.

Чтобы упростить и ускорить эти уроки, вы будете использовать готовые образы в Azure на протяжении всей книги. В реальности вы, вероятно, захотите настроить их в соответствии с потребностями вашего бизнеса. Для этого во многих случаях приходится создавать собственные образы виртуальных машин. Рабочий процесс создания виртуальных машин и управления ими такой же, как и для образов Azure Marketplace, но зачастую для создания собственных образов требуется много времени на планирование, настройку, обобщение и запись образов.

Попробуйте сейчас

Вот несколько идей, которые следует учитывать при планировании запуска приложений в Azure. Они звучат просто, и в большинстве случаев вы можете принимать эти решения автоматически, не тратя время на анализ. Однако по-прежнему важно понимать потребности приложений, прежде чем начать их сборку и запуск.

- В каких регионах должно выполняться приложение? Есть в определенном регионе высокая концентрация пользователей? Как вы обеспечите избыточность?

  Если вы создаете внутренние приложения, запустите их в регионе Azure, который ближе всего к пользователям. Например, если у вас крупный офис в Хьюстоне, штат Техас (возможно, вам нравятся космические корабли!), запускайте приложения и виртуальные машины Azure в Южно-центральном регионе США.

  Если вы разрабатываете внешние приложения, ожидаете ли вы клиентов из определенных регионов? В такой конфигурации может потребоваться развернуть несколько экземпляров в различных регионах (а также обеспечить высокую доступность). Мы вернемся к этой конфигурации в главе 12.
- Вам нужно изменить много параметров ВМ? Сколько времени требуется, чтобы проверить и утвердить все эти изменения? Какие потребности бизнеса движут ими?

  В традиционных локальных средах много времени зачастую уходит на создание предварительно настроенных образов для развертываний. В облаке постарайтесь свести это время к минимуму. Готовые образы Azure содержат последние обновления системы безопасности. Они заранее тестируются и затем географически реплицируются для ускорения развертывания.

Если вы создаете собственные образы, используйте такие функции, как Общая коллекция образов Azure, для распространения и репликации этих образов по мере необходимости (https://docs.microsoft.com/azure/virtual-machines/windows/shared-image-galleries).

2.1.3 Размеры ВМ

В Azure есть различные семейства размеров ВМ. Эти семейства содержат группы сходных типов виртуального оборудования, предназначенных для определенных рабочих нагрузок. Размеры могут обновляться, когда появляется новое оборудование и рабочие нагрузки, но основные семейства остаются неизменными. Типы семейств:

- *Общего назначения* — отличный вариант для разработки и тестирования или для баз данных и веб-серверов производственных сред с низкой нагрузкой.
- *Оптимизировано для вычислений* — высокопроизводительные ЦП, например серверы приложений производственной среды.
- *Оптимизировано для памяти* — большие объемы памяти, например при запуске крупных баз данных или выполнении задач, требующих масштабной обработки данных в памяти.
- *Оптимизировано для хранилища* — низкая задержка и высокая производительность для приложений, интенсивно использующих диски.
- *ГП* — специальные ВМ для графических карт NVIDIA, если вам нужны графическая отрисовка или обработка видео.
- *Высокопроизводительные вычисления* — много всего! Высокие требования к ЦП, памяти и пропускной способности сети для самых ресурсоемких рабочих нагрузок.

Насколько большую ВМ можно создать в Azure? Все течет, все меняется, но на момент написания книги самая большая ВМ, которую можно создать, — «серия Mv2» (часть семейства «Оптимизировано для памяти») со 208 виртуальными ЦП и 5,7 ТиБ памяти. Как думаете, этого хватит для нормального сервера Minecraft?!

Просто знайте, что в Azure количество ВМ, ЦП и памяти ограничено только вашим бюджетом. Скорее всего, вы едва ли создадите ВМ такого размера локально.

При создании виртуальной машины на портале Azure, с помощью интерфейса командной строки или PowerShell необходимо выбрать размер ВМ. Часто для начала работы используют размер виртуальной машины D2s_v3. Вероятно, это чрезмерно для простого веб-сервера из этой главы, но так мы быстрее создадим ВМ и установим необходимые пакеты!

На портале Azure можно фильтровать данные на основе приблизительного размера (например, малого, среднего или большого) или определенного семейства (например, виртуальных машин общего назначения или оптимизированных для памяти). Также показана ежемесячная приблизительная стоимость, чтобы вы могли понять, как дорого будет обходиться каждая ВМ. Обратите внимание на затраты, так как они могут быстро накопиться! Как правило, вы можете определить размер ВМ, после того как она будет включена и запущена, хотя для завершения процесса потребуется выключить и перезагрузить ВМ.

Сокращение затрат на ВМ

Виртуальные машины, созданные по умолчанию, часто предоставляют слишком много ресурсов, но их можно быстро развернуть и начать использовать. Это помогает сократить время, которое потребуется на установку пакетов во время обеденного перерыва.

В реальном мире обратите внимание на требования к памяти, ЦП и хранилищу виртуальных машин. Создавайте ВМ соответствующих размеров. Как и в мире локальных сред, вы можете создать ВМ с завышенным объемом памяти или назначить гораздо больше виртуальных ЦП, чем требуется.

В Azure есть особый тип ВМ: *«серия B»*. ВМ этих размеров используют буферизуемые ЦП и ресурсы памяти, и в этом банке неиспользуемых ресурсов всегда можно взять кредит. Если вы хотите сэкономить свои бонусы от Azure, выполняйте упражнения из книги на ВМ этой серии. Их цены ниже и они отлично подойдут для сценариев, где не требуется много ЦП и ресурсов памяти. Будьте осторожны: в зависимости от размера ВМ серии B, которую вы создаете, у нее может быть меньше ресурсов процессора и памяти, чем, например, у серии D2s_v3, поэтому она будет работать немного медленнее.

2.1.4 Хранилище Azure

Хранилища для ВМ в Azure вполне понятны. Сколько дисков вам нужно, какого объема и типа? Первые два вопроса не относятся к Azure, поэтому мы их пропустим. Доступны следующие типы хранилищ:

- *Твердотельные накопители категории «Премиум»* — высокопроизводительные SSD-диски с низкой задержкой идеально подходят для производственных рабочих нагрузок. Этот тип следует использовать для наилучшей производительности своих приложений.
- *Твердотельные накопители категории «Стандарт»* — используйте стандартные SSD-диски для стабильной производительности на уровне жестких дисков. Этот тип отлично подойдет для разработки и тестовых рабочих нагрузок, а также для дешевых и нетребовательных производственных сред, таких как веб-серверы.
- *Жесткие диски категории «Стандарт»* — это обычные вращающиеся диски, которые идеально подходят для эпизодического доступа к данным, например к архивам или резервным копиям. Этот тип не рекомендуется для запуска рабочих нагрузок приложений.

Чтобы создать быстрый веб-сервер, вам необязательно быть экспертом в специфике хранилищ. Вы узнаете больше в главе 4, в том числе о дисках Ultra, предназначенных только для подключенных дисков данных. А сейчас достаточно знать, что при выборе размера ВМ можно определить используемый тип хранилища.

Используемые вами виртуальные диски, независимо от типа, называются *управляемыми дисками* Azure. Они позволяют создавать ВМ и подключать дополнительные диски с данными, не беспокоясь об используемых учетных записях хранилищ, ограничениях ресурсов или квотах производительности. Кроме того, они автоматически шифруют неактивные данные — вам не нужно настраивать защиту! Опять же, в главе 4 описывается все это и многое другое. На данный момент вы обычно можете позволить Azure создать оптимальный диск на основе выбранного размера ВМ.

Попробуйте сейчас

Чтобы проверить свои знания, ответьте на следующие вопросы:

- Какой тип диска обеспечивает лучшую производительность для большинства рабочих нагрузок?

  SSD-диск категории «Премиум» обычно используется для производственных рабочих нагрузок. Этот тип часто применяется по умолчанию при создании ВМ. SSD-диски категории «Стандарт» хороший второй вариант, а SSD типа Ultra следует использовать только для очень ресурсоемких приложений, которым требуется малая задержка. Несмотря на небольшую экономию со стандартными жесткими дисками, производительность часто такая же высокая, как и в локальных виртуальных средах.
- Какое семейство ВМ оптимально для сервера баз данных?

  Для этого подойдет виртуальная машина, оптимизированная для памяти, так как базам данных часто требуется больше памяти, чем ресурсов ЦП. Всегда старайтесь оценить потребности в ресурсах, а затем отслеживайте производительность после развертывания. Не бойтесь изменять размер ВМ, чтобы добиться нужной производительности.

2.1.5 Виртуальные сети

Кажется очевидным, что для ВМ потребуется сетевое подключение, чтобы пользователи могли получить доступ к вашим приложениям. Для простого веб-сервера нужны и виртуальная сеть, и возможность внешнего подключения. В главе 5 подробно описываются базовые сети Azure, а в главе 9 рассказывается о распределении трафика между несколькими ВМ с помощью подсистем балансировки нагрузки. В главе 11 все становится по-настоящему интересно — в ней рассматривается сервис Azure DNS и глобальная маршрутизация конечных пользователей с помощью диспетчера трафика. Вы не станете сетевым инженером, но узнаете много о сетях Azure!

Начнем с основ, необходимых для этой главы. Виртуальная сеть Azure состоит из тех же базовых компонентов, что и обычная физическая сеть:

- Адресное пространство и маска подсети, например 10.0.0.0/16.
- Одна или несколько подсетей, используемых для разделения внешнего трафика, трафика баз данных и приложений, например
- Сетевые адаптеры виртуальных сетей (NICs), подключающие ВМ к заданной подсети;
- Виртуальные IP-адреса, назначенные виртуальному сетевому адаптеру или балансировщику нагрузки.

Вы можете создать ВМ, подключенную только к виртуальной сети без возможности внешнего подключения, например для серверной базы данных или серверов приложений. Чтобы подключится к этим ВМ для администрирования и обслуживания, можно создать подключение к виртуальной частной сети (VPN) или использовать частную выделенную линию к своему локальному сетевому оборудованию. В Azure эта выделенная линия называется *ExpressRoute*.

Для базового веб-сервера, который вы создадите в этой главе, необходим определенный тип виртуального IP-адреса, а именно публичный IP-адрес. Он назначается виртуальному сетевому адаптеру и позволяет ВМ принимать внешний трафик. Затем вы можете контролировать поток трафика к вашей ВМ с помощью групп безопасности сети (NSG). Вспомните об обычном брандмауэре, который открывает или закрывает различные порты и протоколы; в Azure группы безопасности сети по умолчанию блокируют трафик, разрешая только конкретный его вид, указанный вами. Часто разрешают трафик HTTP или HTTPS на TCP-портах 80 и 443. Также можно включить удаленное управление с помощью протокола удаленного рабочего стола (RDP) или Secure Shell (SSH), но с осторожностью. Вы сделаете позже в этой главе, чтобы увидеть, как подключать и устанавливать некоторые пакеты.

2.2 Создание пары ключей SSH для аутентификации

В практическом задании в конце главы вы создадите то, с чем вы, вероятно, уже знакомы — виртуальную машину Windows Server. Этот тип ВМ использует аутентификацию на основе пароля. Многие приложения в облаке работают в Linux. На самом деле, больше половины виртуальных машин в Azure работают под управлением Linux. Как правило, аутентификация на основе пароля не используется в Linux. Вместо нее применяется SSH и пара открытых ключей. Базовый веб-сервер в этой главе работает под управлением Linux, поэтому вам нужно узнать, как создавать и использовать SSH. Вам не *нужен* опыт работы с Linux для использования облака, но я настоятельно рекомендую вам изучить некоторые основы!

Пары ключей SSH

SSH — это протокол для безопасной коммуникации с удаленными компьютерами и самый популярный способ входа в ВМ Linux. Это напоминает соединение по протоколу удаленного рабочего стола (RDP) на ВМ Windows, но в Linux сеанс SSH обычно выполняется в консоли. Используя шифрование с открытым ключом, вы можете проверить подлинность на удаленной ВМ Linux с помощью пары цифровых ключей.

У пары ключей SSH есть две части: открытый ключ и закрытый ключ. Открытый хранится на вашей ВМ Linux в Azure. Копия закрытого ключа хранится у вас. При входе в ВМ Linux открытый ключ на удаленной ВМ сопоставляется с локальным закрытым. Если пары ключей совпадают, вы входите в ВМ. Это не все аспекты, но в целом шифрование с открытым ключом — отличное средство для подтверждения личности.

Рекомендую вам выработать привычку входа в ВМ Linux с помощью ключей SSH. Ключи SSH гораздо надежнее паролей, поскольку, помимо прочего, не подвержены атакам методом подбора. Всегда обращайте внимание на безопасность, особенно в облаке.

Попробуйте сейчас

Создадим пару открытых ключей SSH с помощью Azure Cloud Shell:

1 Откройте в браузере страницу https://portal.azure.com. Войдите в учетную запись Azure, созданную в главе 1, а затем щелкните значок Cloud Shell в верхней части панели мониторинга, как показано на рисунке 2.2. Вы также можете открыть Cloud Shell напрямую на странице https://shell.azure.com.

Рис. 2.2. Выбор и запуск Cloud Shell на портале Azure с помощью значка оболочки.

2 При первом открытии Cloud Shell потребуется несколько мгновений для создания постоянного хранилища, всегда подключенного к сеансам. Оно позволит вам сохранять и извлекать сценарии, файлы конфигурации и пары ключей SSH. Подтвердите все запросы, чтобы создать хранилище.

3 При необходимости выберите Bash в раскрывающемся меню в левом верхнем углу Cloud Shell. PowerShell также поддерживается, но в этой книге мы сосредоточимся на Bash и Azure CLI.

4 Чтобы создать пару ключей, введите следующую команду:

```
ssh-keygen
```

5 Подтвердите запросы по умолчанию, нажав клавишу ВВОД. Через несколько секунд у вас будет пара открытых ключей SSH, которые можно использовать на всех ваших ВМ! Команда `ssh-keygen` по умолчанию возвращает ключ длиной 2048 битов и использует протокол RCA версии 2. Это достаточно безопасный и рекомендуемый тип для большинства сценариев использования. На рисунке 2.3 показан пример созданной пары ключей SSH в Cloud Shell.

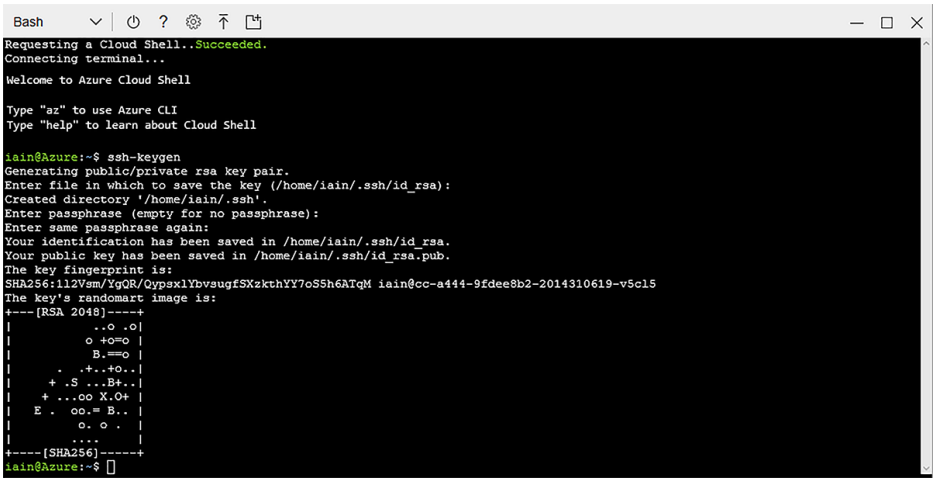


Рис. 2.3. Пара ключей, созданная в Azure Cloud Shell с помощью команды `ssh-keygen`

6 Чтобы просмотреть открытый ключ и использовать его на ВМ, введите следующую команду:

```
cat .ssh/id_rsa.pub
```

7 Выберите выходные данные и скопируйте их в обычный текстовый файл на компьютере. Вы будете использовать этот открытый ключ для создания ВМ в разделе 2.3. На протяжении всей книги мы будем ссылаться на нее в Azure CLI. Обычно не нужно копировать и вставлять весь ключ каждый раз, но будет полезно посмотреть, что происходит на самом деле. Это не сверхсекретная информация, поэтому можно создать и сохранить копию ключа с помощью Блокнота или TextEdit. Внимательно копируйте вывод открытого ключа, поскольку он чувствителен к лишним пробелам и пропуску символов. Вот пример созданного открытого ключа SSH:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDPGaOBsfhJJOHAWAv+RLLR/vdUTzS9HOIj
➥JyzWWLsnu0ESH2M6R+YYPPNXv9X7dmVyM1zCXXEaLucpnyFjevbwPedxTgifyxgCFTgylr1
➥kg7o4EyCTGBGhTA+hSHuhXGXa12KPdKWehsPwHMa6Hs8fbt/in9Z1k2ZAwvbT+LWPcmJgNO
➥FuolIHOsOEeoQQqdXLrGa7NU/3fzSXdT9Y2BT1KLINc4KnwdOuONddLw3iANvK+Gkwax8iK
➥7IicKMoammwvJUCRf+MTEK9pZ84tfsc9qOIAdhrCCLbQhtoWjZpIwYnFk+SNBE8bZZtB8b2
➥vkDFNZlA5jcAd6pUR3tPuL0D iain@cc-a444-9fdee8b2-2014310619-v5cl5
```

СОВЕТ. CLOUD SHELL РАБОТАЕТ В БРАУЗЕРЕ, ПОЭТОМУ СОЧЕТАНИЯ КЛАВИШ ДЛЯ КОПИРОВАНИЯ И ВСТАВКИ МОГУТ СЛЕГКА ОТЛИЧАТЬСЯ ОТ ПРИВЫЧНЫХ ВАМ. ИСПОЛЬЗУЙТЕ CTRL-INSERT И SHIFT-INSERT ВМЕСТО CTRL-C И CTRL-V.

2.3 Создание ВМ в браузере

Теперь, когда вы немного знакомы с теорией виртуальных машин Azure и создали пару ключей SSH, вы готовы создать саму ВМ. Я помогу вам начать, а затем вы настроите ВМ, используя полученные знания, так что будьте внимательны!

Azure CLI и Azure PowerShell — невероятно мощные инструменты, но самая сильная сторона Azure — это скорость создания высококачественного портала. Портал Azure представляет собой графический веб-инструмент, позволяющий увидеть все компоненты вместе и быстро провести визуальную проверку состояния. На портале есть несколько уникальных функций, отсутствующих в других инструментах. Кроме того, он работает очень быстро, поскольку не нужно ничего устанавливать.

Попробуйте сейчас

При создании ВМ в Azure вы можете использовать множество параметров по умолчанию, чтобы ускорить процесс. В этом упражнении мы рассмотрим ресурсы для сети и хранилища, которые Azure создаст, на основе того, что вы узнали в разделе 2.2:

1 На портале Azure (https://portal.azure.com) нажмите кнопку «Создать ресурс» в верхнем левом углу панели управления. Вы увидите популярные ресурсы, в том числе самую последнюю версию Ubuntu с долгосрочной поддержкой (LTS) (на момент написания этой статьи это Ubuntu Server 18.04 LTS).

2 Выберите версию LTS.

Вы также можете выполнить поиск в Marketplace в верхней части окна или просмотреть список высокоуровневых сервисов (таких как вычисления и сети), чтобы понять, что еще доступно для ваших будущих проектов. Выберите Ubuntu Server 18.04 LTS, чтобы следовать одному из предстоящих упражнений по установке компонентов веб-сервера.

3 Создайте группу ресурсов для веб-сервера.

При создании ресурсов в Azure они логически размещаются в определенной вами группе ресурсов. Эти группы обычно содержат схожие ресурсы для приложений. В главе 7 описываются способы планирования и администрирования приложений с помощью групп ресурсов.

Сейчас же я рекомендую вам называть группы ресурсов по главам, так их проще упорядочить. Например, назовите группу ресурсов в этом упражнении `azuremol-chapter2`.

4 Присвойте виртуальной машине имя, например webvm, и выберите ближайший к вам регион. Не беспокойтесь об избыточности инфраструктуры на данный момент.

Изучите параметры образа ВМ, просто чтобы ознакомиться с другими вариантами, но для этого упражнения придерживайтесь Ubuntu Server 18.04 LTS. Размер ВМ по умолчанию подходит для этого упражнения, но опять же, посмотрите, что доступно, как запросить различные размеры аренды и какое оборудование они используют. Посмотрите, как размеры согласуются с семействами ВМ, которые вы изучали ранее в этой главе.

5 Убедитесь, что вы используете аутентификацию с открытым ключом SSH, а затем введите имя пользователя, например azuremol. Вы будете использовать его для входа в ВМ в следующем упражнении.

6 Скопируйте и вставьте открытый ключ SSH, созданный в предыдущем разделе. Опять же, убедитесь, что при копировании и вставке открытого ключа нет лишних пробелов или форматирования. Ключ SSH должен находиться на одной строке. Даже перенос слов в Блокноте может вызывать проблемы! Портал Azure проверяет ключ, прежде чем вы сможете продолжить.

7 Чтобы подключиться к ВМ в следующем упражнении и установить компоненты веб-сервера, откройте n SSH на порту 22.

Открывать SSH на публичной ВМ не очень безопасно. В главе 16 рассказывается, как автоматически открывать и ограничивать доступ, используя JIT-доступ к виртуальным машинам.

Посмотрите на некоторые из других портов, которые вы можете здесь открыть. Часто открываются порты HTTP и HTTPS, и в этой главе вам нужно создать веб-сервер, не так ли? Пока не открывайте эти порты, я хочу познакомить вас с Azure CLI в следующем упражнении, в котором вы разрешите трафик HTTP.

Безопасное подключение с помощью узла-бастиона

В реальных сценариях не следует открывать порты удаленного управления SSH или RDP для Интернета. Правда, не стоит! Следуйте рекомендациям, которые вы соблюдаете в локальной среде (например, подключайтесь только при необходимости и ограничивайте удаленный доступ к определенному набору адресов управления).

(продолжение)

Часто для предоставления удаленного доступа используется узел-бастион или Jumpbox. В такой конфигурации вы не подключаетесь к серверам приложений с ноутбука или настольного компьютера напрямую. Вместо этого вы подключаетесь к выделенному узлу-бастиону, а затем к серверу, которым необходимо управлять. При таком подходе доступ предоставляется ограниченному набору адресов и обеспечивается безопасность удаленного управления.

Бастион Azure (https://docs.microsoft.com/azure/bastion) предоставляет управляемый подход к безопасному удаленному подключению. Вы создаете хост Бастиона Azure в выделенной подсети, а затем используете его для подключения к виртуальным машинам, на которых выполняются приложения. Эти ВМ не обязательно должны быть публичными. Вы можете выполнять все задачи на портале Azure, не открывая сетевые порты для SSH или RDP. Узлом-бастионом управляет система (с точки зрения обновлений системы безопасности и правил групп безопасности сети).

8 Посмотрите на другие параметры конфигурации ВМ для хранилища и сети, чтобы ознакомиться с вариантами. Но на данный момент вы можете оставить все значения по умолчанию.

9 Есть также несколько интересных параметров управления, таких как автоматическое выключение, резервное копирование и диагностика, которые описываются в главах 12 и 13. Сейчас же отключите такие диагностику загрузки и гостевой ОС, так как вам нужно создать и настроить учетную запись хранения, чтобы они работали.

10 Когда все будет готово, проверьте параметры и создайте базовую ВМ.

2.4 Подключение к ВМ и установка веб-сервера

Когда ВМ уже работает, можно войти в нее с помощью ключа SSH, созданного ранее. Затем вы можете начать устанавливать и настраивать веб-сервер в Cloud Shell.

2.4.1 Подключение к ВМ с помощью SSH

В этом разделе мы узнаем, как быстро получить сведения о подключении для ВМ.

Попробуйте сейчас

Если вы не знакомы с Linux, не волнуйтесь! Выполните эти несколько шагов, чтобы войти в свою ВМ:

1 На портале Azure выберите раздел «Виртуальные машины» на панели навигации в левой части экрана. Создание ВМ из предыдущего упражнения занимает пару минут, поэтому нажимайте кнопку «Обновить», пока состояние ВМ не изменится на *Выполняется*. Когда все будет готово, выберите ВМ и нажмите кнопку «Подключить», как показано на рисунке 2.4.

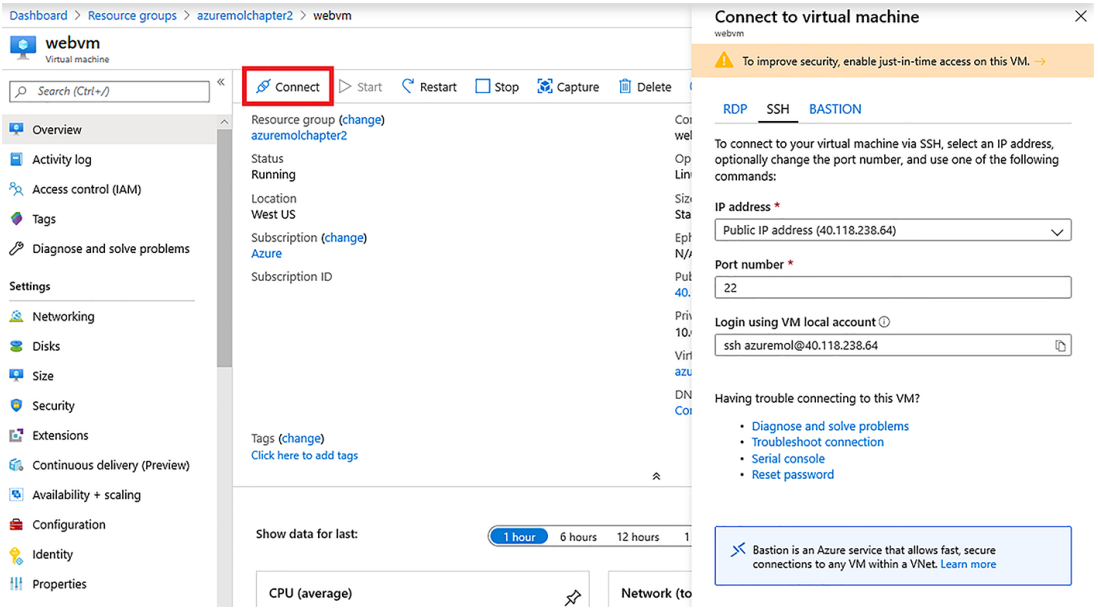


Рис. 2.4. Выберите свою ВМ на портале Azure, а затем нажмите кнопку «Подключиться», чтобы создать сведения об SSH-подключении.

На ВМ Linux вы увидите команду SSH, включающую ваше имя пользователя и публичный IP-адрес. Скопируйте эту команду подключения, например `ssh azuremol@104.209.208.158`.

В случае с ВМ Windows при нажатии на кнопку «Подключиться» на ваш компьютер загружается файл RDP-подключения с предварительно заполненным IP-адресом ВМ.

2 При необходимости повторно откройте Cloud Shell. Если планируете переключаться между Cloud Shell и порталом, вы можете свернуть Cloud Shell, оставляя доступ к оболочке в фоновом режиме.

3 Вставьте команду SSH в Cloud Shell, а затем нажмите клавишу ВВОД. Созданный вами ранее ключ SSH автоматически используется для проверки подлинности.

При первом подключении к ВМ с помощью SSH вам будет предложено добавить ВМ в список надежных узлов. Это еще один уровень безопасности, который обеспечивает SSH. Если кто-либо попытается перехватить трафик и направить вас на другую удаленную ВМ, ваш локальный клиент ВМ узнает об этом изменении и заблаговременно оповестит вас.

Подтвердите запрос на хранение подключения к удаленной ВМ. На рисунке 2.5 показан процесс SSH-подключения в Azure Cloud Shell.

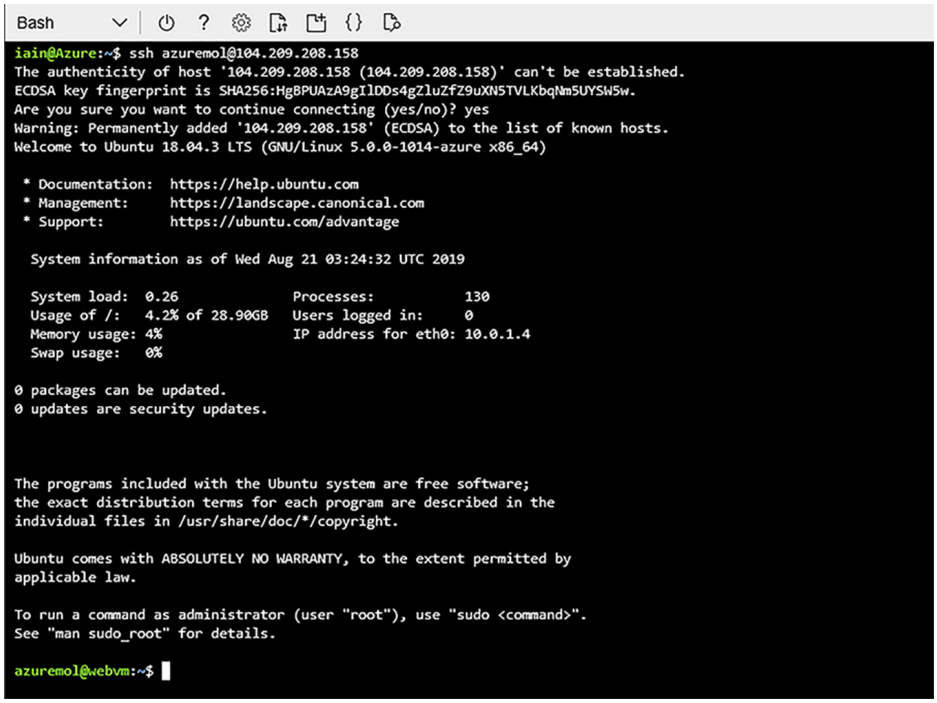


Рис. 2.5. Используйте строку подключения на портале Azure, чтобы создать SSH-подключение к своей ВМ из Cloud Shell.

На данный момент вы либо вдали от дома, либо запрос Linux полностью иностранный. Не волнуйтесь. Вам не нужно знать множество команд Linux, к тому же мы будем разбирать каждую новую команду в дальнейшем. Тем не менее, настоятельно рекомендую вам хотя бы частично овладеть администрированием в Linux. Многие облачные решения основаны на системах Linux, а приложения все чаще разрабатываются и управляются с помощью контейнеров и микросервисов. Если вы администратор Windows со стажем, добро пожаловать! Я кое-что приберег для вас на конец главы, так что оставайтесь с нами.

2.4.2 Установка веб-сервера

Создать виртуальную машину? Готово. Подключиться к ВМ с помощью SSH? Готово. Теперь можно устанавливать пакеты для веб-сервера и приготовиться к испытаниям.

Azure поддерживает много различных *дистрибутивов*. Поэтому инструменты управления пакетами и расположение файлов конфигурации немного отличаются в зависимости от дистрибутива. В этой книге мы будем использовать Ubuntu, поскольку это один из самых популярных и хорошо задокументированных дистрибутивов Linux для облачных вычислений. Если вы немного застряли, вас всегда выручит обширная документация по адресу https://help.ubuntu.com. Если вы хотите использовать другой удобный для вас дистрибутив — не стесняйтесь! В противном случае пользуйтесь Ubuntu.

Попробуйте сейчас

Перейдите из SSH-сеанса к ВМ и установите пакеты веб-сервера с помощью APT:

1. В Ubuntu пакеты устанавливаются с помощью Advanced Packing Tool (APT) — сверхмощного инструмента управления пакетами, который автоматически устанавливает любые дополнительные пакеты, которые необходимы. Просто выберите «Установить веб-сервер», и APT установит все требуемые компоненты.

   Для этого примера установите веб-стек LAMP. Это, вероятно, самый распространенный набор веб-компонентов: Linux, Apache (веб-сервер), MySQL (сервер баз данных) и PHP (язык веб-программирования):

   ```
   sudo apt update && sudo apt install -y lamp-server^
   ```

   Первая команда обновляет доступные пакеты. Это рекомендуется, чтобы установить самые последние и лучшие из них. Затем вы запускаете следующую команду с символом `&&`. Почему бы просто не ввести ее с новой строки? `&&` выполняет следующую команду только в случае успешного выполнения предыдущей. Например, если при получении последних версий пакетов посредством команды apt отсутствовало сетевое подключение (уж поверьте, я знаю, что у вас должно быть сетевое подключение, чтобы вы вообще смогли подключиться!), нет смысла выполнять команду `install`.

   Если команда update выполнена успешно, команда apt определяет необходимые дополнительные пакеты и начинает установку `lamp-server`. Для чего нужен знак вставки в конце строки (^)? Он позволяет команде `apt` установить весь набор пакетов, формирующих сервер LAMP, а не только один пакет под названием `lamp-server`.

2. Программа установки может запросить у вас пароль или по умолчанию воспользуется пустым паролем MySQL. Это небезопасно, и для реальной производственной среды следует указать надежный пароль. В главе 15 мы станем настоящими профи и сохраним надежный, безопасный пароль в решении Azure Key Vault, автоматически встроенном в этот мастер установки MySQL.

   Установка всех пакетов для вашего веб-стека LAMP займет около минуты, а затем можно двигаться дальше.

3. Введите команду `exit`, чтобы выйти из ВМ и вернуться в командную строку Cloud Shell.

Вот и все! Веб-сервер уже запущен, но вы пока что не сможете открыть его в браузере. Нам понадобится открыть ВМ для веб-трафика.

2.5 Открытие доступа к ВМ для веб-трафика

Ваш веб-сервер уже работает, но если ввести публичный IP-адрес ВМ в браузере, вы не сможете загрузить веб-страницу. Зачем? Помните группы безопасности сети, кратко рассмотренные в разделе 2.1.5? При создании ВМ для вас была создана группа безопасности сети, Было добавлено правило, разрешающее удаленное управление — в данном случае SSH.Остальная часть виртуальной машины заблокирована. Чтобы пользователи получили

доступ к вашему веб-серверу через Интернет, необходимо создать правило в группе безопасности сети, разрешающее веб-трафик. В противном случае никто не сможет заказать пиццу!

2.5.1 Создание правила, разрешающего веб-трафик

В этом разделе мы устроим небольшую встряску и создадим правило для веб-трафика с помощью Azure CLI. Вы могли бы открыть этот порт HTTP на портале при создании ВМ, но тогда вы бы упустили половину веселья!

Интерфейс Azure CLI доступен в Cloud Shell. Ничего устанавливать не придется. В главе 5 мы подробнее остановимся на виртуальных сетях и группах безопасности сети, а сейчас оценим скорость и мощь Azure CLI, введя в нем всего одну команду.

Попробуйте сейчас

Откройте Azure Cloud Shell и выполните эти шаги, чтобы увидеть Azure CLI в действии:

1. Если окно Cloud Shell закрыто, откройте его на портале Azure. Убедитесь, что загружается Bash, а не PowerShell. При необходимости переключитесь на Bash-версию.
2. Чтобы просмотреть Azure CLI и установленные модули, введите команду `az --version`. На экране отобразится список модулей и номера версий. Главное преимущество оболочки Cloud Shell — она всегда использует самую последнюю и лучшую версию.

ПРИМЕЧАНИЕ. НАВЕРНЯКА ВЫ ЗАМЕТИЛИ, ЧТО КОМАНДА ВЫВЕЛА ИНФОРМАЦИЮ О ВЕРСИИ PYTHON. ПОЧЕМУ ЭТО ТАК ВАЖНО? PYTHON — МОЩНЫЙ И ПОПУЛЯРНЫЙ ЯЗЫК ПРОГРАММИРОВАНИЯ. НА НЕМ НАПИСАН AZURE CLI — КРОСС-ПЛАТФОРМЕННЫЙ ИНТЕРФЕЙС, ДОСТУПНЫЙ ДЛЯ ЛОКАЛЬНОЙ УСТАНОВКИ НА ЛЮБОМ КОМПЬЮТЕРЕ И ЗАМЕНЯЮЩИЙ CLOUD SHELL. АКТИВНО УЧАСТВУЯ В СООБЩЕСТВЕ РАЗРАБОТЧИКОВ ПО С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ, КОРПОРАЦИЯ MICROSOFT СДЕЛАЛА AZURE CLI ДОСТУПНЫМ НА GITHUB, ЧТОБЫ ВСЕ ЖЕЛАЮЩИЕ МОГЛИ РАЗРАБАТЫВАТЬ ПРОГРАММЫ, ВНОСИТЬ ПРЕДЛОЖЕНИЯ ИЛИ СООБЩАТЬ О ПРОБЛЕМАХ (https://github.com/Azure/azure-cli).

3. Чтобы открыть порт, укажите имя ВМ, группу ресурсов, а также номер порта. Для веб-трафика откройте порт 80. Введите группу ресурсов (`-g`) и имя ВМ (`-n`), указанное при ее создании:

```
az vm open-port -g azuremolchapter2 -n webvm --port 80
```

2.5.2 Обзор работы веб-сервера

Теперь, когда вы открыли порт для ВМ, давайте проверим доступ к виртуальной машине в браузере.

1. Если вы перешли на другую страницу, выберите свою ВМ на портале Azure. Публичный IP-адрес указан в верхнем правом углу страницы обзора ВМ.
2. Выберите и скопируйте адрес.

3 В браузере откройте новую вкладку или окно и вставьте публичный IP-адрес в строку подключения. По умолчанию загрузится веб-сайт Apache, как показано на рисунке 2.6. Конечно, это не похоже на магазин для доставки пиццы, но фундамент заложен, и нужен только код для создания вашего приложения!

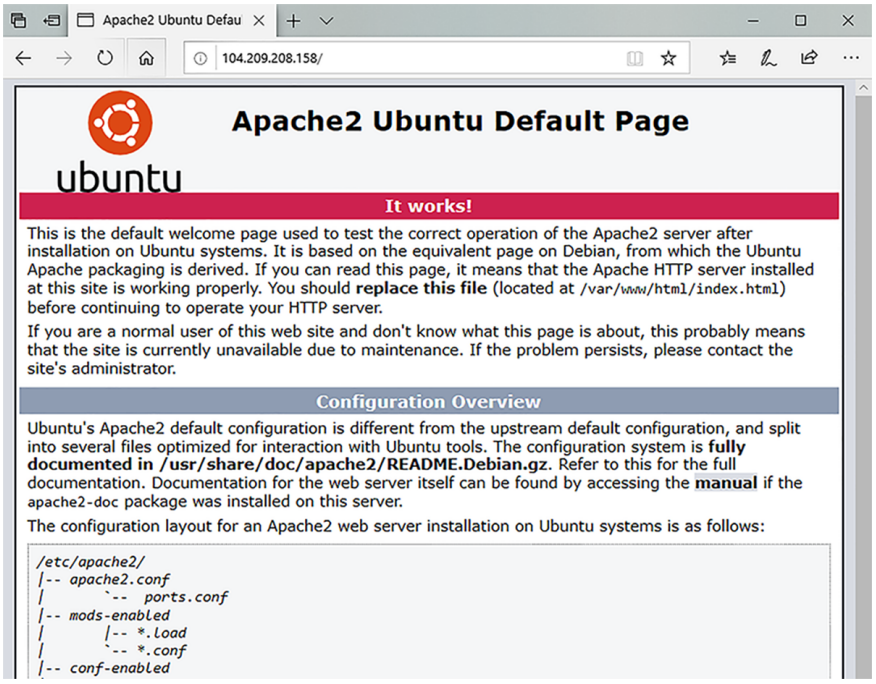


Рис. 2.6. Чтобы увидеть веб-сервер в действии и просмотреть стандартную страницу Apache 2, введите публичный IP-адрес в браузере.

2.6 Лаборатория: создание ВМ Windows

В предыдущих разделах мы выполнили все шаги по установке стека LAMP на ВМ Ubuntu Linux. Это стандартная платформа для веб-сайтов, но и поддержка Windows никогда не помешает! Возможно, группы разработчиков или ваши коллеги, принимающие бизнес-решения, хотят работать, например, с .NET. Хотя вы можете запустить .NET Core на ВМ Linux, не позволяйте языку программирования определять ваши приоритеты.

Используя знания, полученные в пошаговом примере, попробуйте создать ВМ, выполняющую сервисы IIS. Несколько советов:

- Вам понадобится ВМ, выполняющая Windows Server 2019.
- Вы используете RDP вместо SSH, поэтому будьте готовы к небольшим отличиям при подключении.
- В диспетчере сервера найдите параметр «Добавить роли и функции».
- Необходимо установить веб-сервер (IIS).
- Не забудьте открыть сетевой порт для трафика HTTP на TCP-порту 80. Для этого можно использовать портал.

2.7 Очистка ресурсов

Когда вы создаете ресурсы в Azure, запускается выставление счетов. Вы платите поминутно, поэтому не оставляйте ресурсы, например ВМ, запущенными без необходимости. Есть 2 способа остановить выставление счетов за выполнение ВМ:

- *Отмена выделения ВМ*. Нажмите кнопку «Остановить» на портале, чтобы остановить ВМ, отменить ее выделение и освободить все задействованные вычислительные и сетевые ресурсы.
- *Удаление ВМ*. Здесь все ясно. Если в Azure ничего нет, то и платить не за что. Перед тем как удалять ВМ, убедитесь, что она больше не нужна вам. В Azure нет кнопки «Отмена»!

Рекомендую создавать группу ресурсов перед каждой разработкой приложения в Azure. Как раз это вы и делаете, выполняя упражнения в книге. Если вы назовете группы ресурсов по главам, например `azuremolchapter2`, вам будет легче отслеживать ресурсы и понимать, что удалять. Это сделает очистку немного проще, поскольку вы сможете удалять целую группу ресурсов в конце каждой главы. Выберите пункт «Группы ресурсов» в меню навигации слева, откройте каждую группу ресурсов, созданную вами в этой главе, а затем выберите «Удалить группу ресурсов», как показано на рисунке 2.7. Для подтверждения вы получите запрос об имени группы ресурсов.

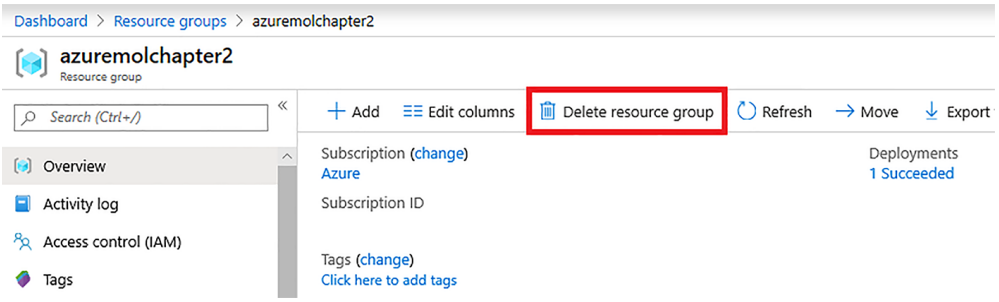


Рис. 2.7. В целях экономии удаляйте ненужные группы ресурсов.

Если вы привыкнете удалять ненужные ресурсы, вам вполне хватит бонусов Azure на изучение этой книги. Как минимум отменяйте выделение ВМ в конце каждого урока, чтобы продолжить на следующий день и остановить выставление счетов.

2.8 Хьюстон, у нас проблема

Иногда вы будете сталкиваться с проблемами в Azure. Да, я это сказал. Как правило, на платформе Azure удобно устранять неполадки при создании ресурсов:

- Azure CLI или Azure PowerShell сообщает о проблемах при выполнении команд, поэтому вы заметите, если что-то пойдет не так. Как правило, Azure PowerShell использует приятный и спокойный текст красного цвета, чтобы привлечь ваше внимание.

- Azure CLI тщательнее шифрует уведомления, поскольку обычно включает фактические ответы на базовые вызовы REST API от сервера. При первых попытках разобраться в происходящем вы можете пережить и взлеты, и падения. Полезная сторона REST-ответов: вы можете скопировать и вставить сообщения об ошибках в свой любимый поисковик и в большинстве случаев получить исчерпывающие рекомендации по устранению неполадок.

Уже хотите отдохнуть от REST? Мы только начали!

Когда вы открываете веб-страницу в браузере, ваш компьютер взаимодействует с веб-сервером по протоколу HTTP. Я уверен, что вы неоднократно видели на сайтах сообщения об ошибках 404. Это означает, что веб-страница не найдена. Другие распространенные ошибки: 403, если у вас нет разрешений на просмотр веб-страницы, и 500, если сервер обнаружил ошибку.

Даже когда все в порядке, браузер получает сообщения с кодом 200, если страница загружена без ошибок, или сообщения с кодом 301 при переадресации. Вам не нужно понимать и отслеживать все эти коды; это просто стандартный метод HTTP для коммуникации между компьютерами.

Я уже рассказывал о создании и контроле ресурсов с помощью веб-портала, CLI или PowerShell. Доступ ко всем сервисам Azure обеспечивают интерфейсы программирования (API) передачи репрезентативного состояния (REST).

Если вы еще не знаете, REST API — это стандартизированный метод предоставления сервисов через HTTP. Вы используете HTTP-запросы, например `GET` и `POST`, чтобы получить сведения или внести изменение. Когда платформа принимает и обрабатывает запрос, вы получаете сообщение о состоянии. В Azure есть четко определенный набор REST API.

Вам не нужно понимать, что все это значит. Просто знайте, что сообщение об ошибке не всегда выводится в понятном и удобном для человека формате. Иногда вы получаете от REST API необработанный HTTP-ответ, требующий расшифровки. Опять же, вставьте эту ошибку в строку поиска любимой поисковой системы. Высока вероятность того, что кто-то уже столкнулся с проблемой и описал, что пошло не так и что вам нужно исправить.

Самые распространенные проблемы возникают при подключении к ВМ. Возможно, вы подключались для удаленного администрирования с помощью SSH или RDP или пытались открыть свои приложения в браузере либо клиенте для настольных ПК. Эти проблемы часто связаны с сетью. Отложу пока все претензии к сетевикам до главы 5 и приведу вам несколько простых проверок:

- Вы можете подключится к любой другой ВМ Azure или приложениям, запускаемым в Azure? Если нет, значит локальные параметры вашей сети блокируют доступ.

  Если вам удалось подключится к другим ресурсам Azure, убедитесь, что вы открыли правила безопасности сети, о которых я говорил в разделе 2.5. Эти правила подробно описываются в главе 5.

- В случае проблем с аутентификацией выполните следующие действия:
  - Убедитесь, что у вас есть правильные ключи SSH. Azure уведомит вас о недопустимом открытом ключе при создании ВМ, но если у вас их несколько, убедитесь, что используете правильный!
  - В случае проблем с RDP попробуйте подключиться к localhost\<имя пользователя> и ввести свой пароль. По умолчанию большинство RDP-клиентов предоставляют локальные учетные данные или сетевые учетные данные, непонятные для вашей ВМ.

3 Веб-приложения Azure

В главе 2 вы создали ВМ и вручную установили пакеты, чтобы запустить простой веб-сервер. Вы могли бы создать интернет-магазин пиццы с помощью этой ВМ, если бы изголодались по решительным действиям. Один из самых популярных вариантов использования ВМ Azure ВМ — запуск веб-приложений. Веб-приложения являются удобной рабочей нагрузкой для ВМ. Но удобство — это плюс, если вам также нравится управлять всеми этими ВМ и обслуживать их. Знаете, все эти веселые штуки типа обновлений программ и системы безопасности, централизованного ведения журнала и отчетов о соответствии. Хотите получить все возможности безопасного веб-сервера для запуска своих веб-приложений? Включая автоматическое масштабирование согласно собственным требованиям, но без необходимости создавать и контролировать все эти ВМ? Позвольте представить вам сервис «Веб-приложения Azure».

В этой главе мы сравним два подхода к ВМ и веб-серверам: «Инфраструктура как услуга» (IaaS) и «Платформа как услуга» (PaaS). Вы изучите преимущества сервиса «Веб-приложения Azure», создавая собственное веб-приложение, а также поймете процесс его разработки и выпуска в производственной среде. Затем вы узнаете, как автоматически развернуть веб-приложение из системы управления версиями, например из GitHub. Этот рабочий процесс показан на рисунке 3.1. Веб-приложения Azure позволяют развертывать и запускать веб-приложения в Интернете интернет-магазин пиццы за считанные минуты без необходимости настраивать ВМ и пакеты веб-серверов.

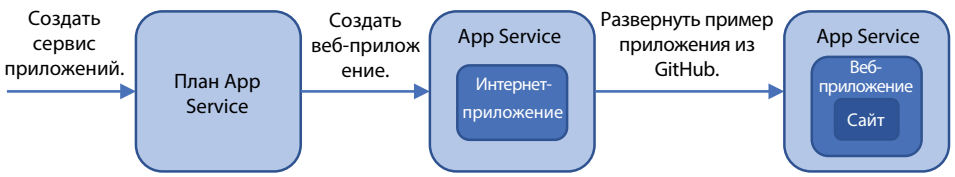


Рис. 3.1. В этой главе вы создадите сервисный план приложений и основное веб-приложение, а затем выполните развертывание веб-сайта из GitHub.

3.1 Обзор сервиса «Веб-приложения Azure» и основные понятия

Благодаря сервису «Веб-приложения Azure» вы начинаете погружаться в удивительный мир PaaS-решений. Если вы думаете, что облачные вычисления полностью посвящены ВМ, вам следует немного перезагрузиться. В начале книги я говорил о покупке вычислительных ресурсов и акценте на ваших приложениях и клиентах. По мере вашего перехода от решений IaaS, например от ВМ, к таким решениям PaaS, как веб-приложения, ваши программы и клиенты становятся центром внимания.

Чтобы запускать приложения на ВМ IaaS, требуются контроль ОС, обновления приложений, правила безопасности и трафика, а также настройка всей системы. С сервисом «Веб-приложения» вы загружаете свое веб-приложение, и все эти задачи выполняются за вас. Теперь вы можете сосредоточится на улучшении приложений для своих клиентов или повышении надежности с помощью масштабирования и контроля трафика.

Означает ли это, что вам вообще не следует запускать ВМ для размещения веб-приложений? Скорее, нет. Есть веские причины для самостоятельного запуска и настройки всего стека приложений, например если вам нужна поддержка конкретных приложений или языковой среды выполнения. Однако сервис «Веб-приложения» зачастую облегчает их запуск.

3.1.1 Поддержка языков и сред

Какие языки программирования можно использовать в сервисе «Веб-приложения»? Самые разные! Две главные платформы для запуска этого сервиса: Windows и Linux. Веб-приложения .NET Core, Node.js, Python, Java, Ruby и PHP можно запускать на экземплярах веб-приложений Windows и Linux. В Windows также можно запустить полную платформу .NET Framework. А если вы хотите быть круче и запускать свои веб-приложения в контейнерах, к вашим услугам также сервис «Веб-приложения для контейнеров». Он позволяет запускать собственные контейнеры Docker для Linux. Мы подробно рассмотрим контейнеры и Docker в главе 19, а пока достаточно понимать, что сервис «Веб-приложения» охватывает все варианты!

Когда нецелесообразно использовать сервис «Веб-приложения»? Не все языки приложений поддерживаются сервисом «Веб-приложения». Например, вы хотите помучить себя, запуская веб-приложение на языке Perl. В таком случае вы, вероятно, вернетесь к запуску на собственной ВМ IaaS, поскольку сервис «Веб-приложения» не поддерживает Perl. Но в целом он поддерживает самые распространенные языки программирования, на которые вы ориентируетесь. Кстати, вам не помешает подыскать версию приложения посвежее. Вместо той, на языке Perl.

Сервис «Веб-приложения» не только поддерживает разные языки, но еще и различные их версии. Возьмем, к примеру, PHP. Как правило, вы можете выбрать 3 или 4 версии PHP для оптимальной поддержки приложения. А самое интересное, в отличие от своей ВМ IaaS, вам не нужно волноваться о том, чтобы базовый веб-сервер поддерживал их все. Python — еще один пример различий между версиями 2.7 (стабильной) и 3.6 (а также последующими), как показано на рисунке 3.2.

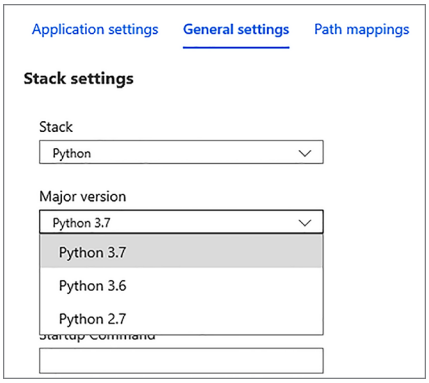


Рис. 3.2. Выберите конкретную версию языка в параметрах приложений сервиса «Веб-приложения».

Кроме того, в сервисе «Веб-приложения» всегда установлены последние исправления безопасности. Но не ждите, что старые версии PHP или Python будут поддерживаться бесконечно. На определенном этапе поддержка устаревших версий будет прекращена. Опять же, если для вашего приложения необходима более старая версия языка, возможно, вам стоит вернуться к использованию своей ВМ IaaS. Но если вам нужно запускать старую версию данного языка, только чтобы поддерживать устаревшее приложение, не увязайте в постоянном обслуживании. Всегда пытайтесь перевести устаревшие приложения на современные поддерживаемые платформы.

3.1.2 Промежуточное хранение различных версий в слотах развертывания

Слоты развертывания обеспечивают среду с различными версиями для вашего приложения. Вы можете отправлять новые версии своего приложения в слот развертывания и запускать их с помощью переменных среды или через подключения к базе данных без влияния на действующий сайт. Если вы довольны внешним видом и функциями в слоте развертывания, можете мгновенно переключить сайт на эту версию. Затем предыдущая версия сайта архивируется в своем слоте развертывания, но вы также можете вернуть ее в производственную среду при необходимости.

Число доступных слотов развертывания зависит от уровня вашего веб-приложения. Дополнительные слоты развертывания позволяют разным разработчикам размещать и тестировать свои обновления в нескольких промежуточных версиях.

3.1.3 Планы App Service

Сервис «Веб-приложения» — часть широкого семейства App Service в Azure. Azure App Service также включает в себя «Мобильные приложения», «Приложения API» и Logic Apps. Все сервисы, кроме Logic Apps, доступны во всех регионах Azure. Вот отличный ресурс для проверки доступности сервиса Azure по регионам — https://azure.microsoft.com/regions/services. Многие сервисы доступны в глобальном масштабе.

Если вам нужно создать ресурс сервиса приложений, например веб-приложение, вы создаете сервисный план или используете существующий. Сервисный план определяет число доступных вам ресурсов, объем автоматизации для масштабирования и архивирования вашего веб-приложения, а также уровень доступности вашего сайта со слотами промежуточного развертывания и с

диспетчером трафика (это средство для направления трафика на экземпляр, который ближе всего к пользователю, что мы рассмотрим в главе 11). Как обычно, вы получаете то, за что заплатили. Вам следует определить объем требуемых ресурсов и необходимые дополнительные функции на основе ваших приложений и бизнес-потребностей. Уровень каждого сервиса основан на функциях предыдущих уровней; как правило, добавляются объем хранения и доступные ресурсы.

4 основных уровня сервисного плана:

- *«Бесплатный» или «Общий»* — использует общую инфраструктуру, предлагает минимальный объем хранилища и не содержит параметров для развертывания различных промежуточных версий, маршрутизации трафика или резервных копий. «Общий» уровень позволяет вам эксплуатировать личный домен, но взимает за него плату.
- *«Базовый»* — предоставляет выделенные вычислительные ресурсы для вашего веб-приложения и позволяет использовать SSL и вручную масштабировать количество запущенных экземпляров веб-приложения. «Бесплатный», «Общий» и «Базовый» уровни предоставляют эффективную среду для тестирования сервиса «Веб-приложения», но я бы не рекомендовал запускать какие-либо производственные рабочие нагрузки или рабочие нагрузки разработки. Производительность не будет ограничивающим фактором, но вы упустите ряд автоматизированных функций, таких как резервное копирование и масштабирование.
- *«Стандартный»* — добавляет ежедневные резервные копии, автоматическое масштабирование экземпляров веб-приложений, слоты развертывания, а также позволяет вам маршрутизировать пользователей с помощью диспетчера трафика. Этот уровень подходит для приложений с низкой нагрузкой спросом или для сред разработки, в которых не требуется множество резервных копий или слотов развертывания.
- *«Премиум»* — повышает частоту резервного копирования, увеличивает объем хранилища, предоставляет больше слотов развертывания и параметров масштабирования экземпляров. Этот уровень идеально подходит для большинства рабочих нагрузок.

Случаи изоляции

В таких PaaS-решениях, как сервис «Веб-приложения», инфраструктура намеренно абстрагируется. Некоторые уровни сервисного плана предполагают, что веб-приложения запускаются на всех общих платформах доступных ресурсов. Это вовсе не означает, что веб-приложения небезопасны и другие пользователи смогут просматривать ваши личные данные. Но чтобы не нарушать нормативных требований, вам придется запускать приложения в контролируемой изолированной среде. Для этого пригодятся *среды сервиса App Service* — изолированные среды, позволяющие запускать экземпляры сервиса приложений как веб-приложения в сегментированной части ЦОД Azure. Вы контролируете входящий и исходящий сетевой трафик и можете внедрять брандмауэры и создавать подключения к своим локальным ресурсам через виртуальные частные сети (VPN).

Все эти компоненты инфраструктуры по-прежнему значительно абстрагированы по средам сервиса приложений. Тем не менее, этот подход обеспечивает отличный баланс между гибкостью решений PaaS и углубленным контролем трафика сетевых подключений.

У вас немало возможностей на уровнях «Бесплатный» и «Базовый», хотя для рабочих нагрузок лучше использовать уровни «Стандартный» и «Премиум». В примере в этой главе используется уровень «Стандартный», чтобы вы смогли увидеть все доступные функции. Используя сервис «Веб-приложения» со своими приложениями, вы можете выбрать нужные вам функции и оптимальный уровень сервисного плана.

3.2 Создание веб-приложения

Изучив немного теории, давайте посмотрим на сервис веб-приложений в действии. Для запуска приложения необходимо выполнить несколько действий. Во-первых, создайте простое веб-приложение и откройте стандартный сайт в браузере. Затем, найдите в GitHub образец веб-страницы и отправьте его в Azure. Возможно, ваши веб-разработчики начали создавать внешний интерфейс для интернет-магазина пиццы, и у вас уже есть простой сайт для загрузки.

> ПРИМЕЧАНИЕ. ЕСЛИ ВЫ НИКОГДА НЕ ИСПОЛЬЗОВАЛИ GIT, НЕ ВОЛНУЙТЕСЬ. НА ДАННОМ ЭТАПЕ ВАМ НЕ НУЖНО ПОНИМАТЬ, ЧТО ДЕЛАЕТ GIT, А В КОНЦЕ ГЛАВЫ ВЫ СМОЖЕТЕ НЕМНОГО ПОЭКСПЕРИМЕНТИРОВАТЬ. НАУЧИТЕСЬ РАБОТАТЬ С GIT ЗА МЕСЯЦ РИКА УМАЛИ (HTTPS://WWW.MANNING.COM/BOOKS/LEARN-GIT-IN-A-MONTH-OF-LUNCHES) — ОТЛИЧНОЕ РУКОВОДСТВО ПО НАЧАЛУ РАБОТЫ С GIT, ЕСЛИ ВЫ ХОТИТЕ УЗНАТЬ НЕМНОГО БОЛЬШЕ. ЕГО МОЖНО ЧИТАТЬ БЕСПЛАТНО НА ПЛАТФОРМЕ MANNING LIVEBOOK.

3.2.1 Создание простого веб-приложения

Как и в главе 2, я буду предоставлять общие рекомендации на этом пути, однако вам предстоит решить, сможете ли вы применить теорию к средам выполнения приложений и планам App Servic для создания веб-приложения. Если вы не уверены в некоторых параметрах, можно с уверенностью принять значения по умолчанию.

PaaS, а не IaaS

Это веб-приложение — новый ресурс, отделенный от виртуальных машин, подобных той, которую вы создали в главе 2, описывающей модель IaaS для создания и запуска веб-приложений. Подход PaaS — это сервис «Веб-приложения». Между этими 2 подходами нет взаимосвязи. Фактически, если вы последовали совету в главе 2 и удалили ВМ, то это веб-приложение запускается вообще без ВМ в вашей подписке Azure!

Попробуйте сейчас

Чтобы создать веб-приложение, выполните приведенные ниже действия:

1. Откройте https://portal.azure.com в браузере и войдите в свою учетную запись Azure.
2. На портале нажмите кнопку «Создать ресурс» в верхнем левом углу панели.
3. Выберите «Веб-приложения» из списка доступных для создания ресурсов, а затем нажмите кнопку «Веб-приложение».

4 Чтобы все было так же четко и красиво, как и в главе 2, рекомендую создать выделенную группу ресурсов для веб-приложения, например azuremolchapter3.

5 В качестве имени веб-приложения введите глобально уникальное имя. Оно должно быть уникальным в глобальном масштабе, поскольку оно создает URL-адрес вашего веб-приложения в формате http://*<имя>*.azurewebsite.net. Если вам интересно: да, здесь можно применить имя личного домена. Пока же используйте адрес azurewebsites.net.

6 Вы отправите базовый HTML-код, а не контейнер Docker, но посмотрите в различные доступные стеки сред выполнения. Этот параметр можно изменить после создания веб-приложения, но сейчас выберите среду выполнения ASP.NET, которая работает в Windows.

7 Позвольте Azure автоматически создать план App Service, но измените размер на «S1 Standard». Этот уровень обеспечивает вас всеми основными функциями без лишних ресурсов для демонстрационного веб-сайта. В реальных развертываниях на этом шаге можно вручную создать и настроить собственные планы App Service или выбрать существующий план.

8 Когда все будет готово, проверьте данные и создайте свое первое веб-приложение.

Создание сервиса приложений займет несколько секунд. Когда будете готовы, найдите и выберите сервисы приложений на панели навигации в левой части экрана, а затем выберите веб-приложение из списка. В окне «Обзор» своего веб-приложения выберите для него URL-адрес, например https://azuremol.azurewebsites.net..

При выборе URL-адреса веб-приложения в браузере открывается новое окно или вкладка. По умолчанию загружается страница веб-приложения, как показано на рисунке 3.3. Все еще не выглядит как пицца. . . .

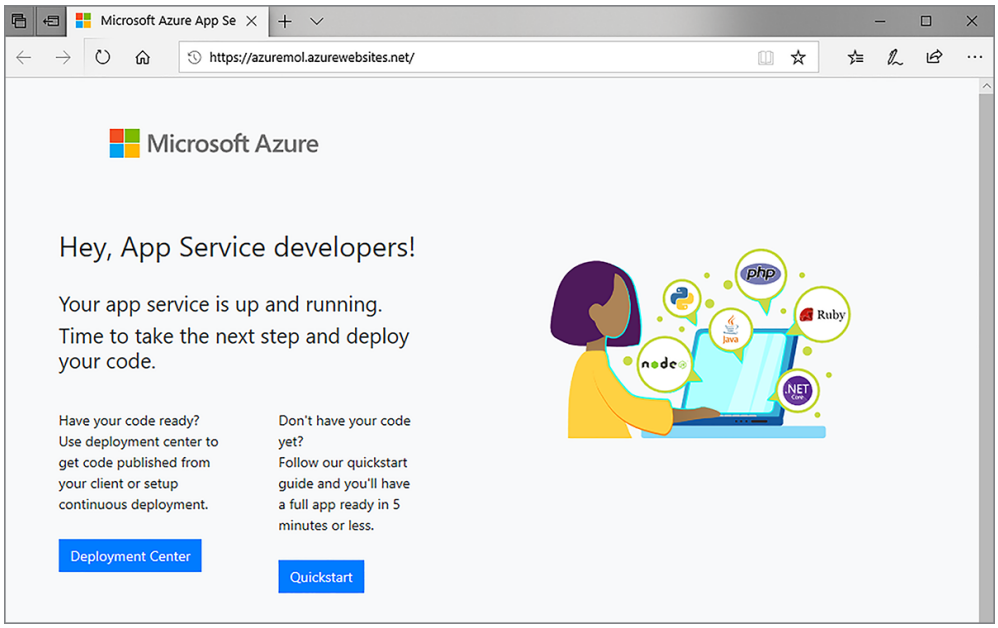


Рис. 3.3. Чтобы увидеть стандартную страницу веб-приложения в действии, откройте браузер и перейдите по URL-адресу своего сайта.

3.2.2 Развертывание образца сайта в формате HTML

У вас уже есть веб-приложение в Azure, но это всего лишь унылый стандартный веб-сайт. Как же создать настоящий веб-сайт в Azure? Один из наиболее распространенных кросс-платформенных способов — использование Git. Большинство разработчиков и команд используют систему управления версиями. Вместо того, чтобы хранить файлы на компьютере и сохранять изменения по ходу работы, системы управления версиями отслеживают изменения и позволяют вам работать вместе с другими пользователями. Вы можете создавать тестовые выпуски, не влияющие на производственный код, и возвращаться к ранним версиям в случае проблем. Git — одна из самых популярных систем управления версиями, а GitHub — облачный сервис, позволяющий совместно использовать и изменять код в мировом масштабе. Корпорация Microsoft приобрела GitHub в июне 2018 г., но это вовсе не означает, что вы должны обязательно использовать GitHub с Azure или наоборот. Все примеры из этой книги доступны на GitHub.

Для этого примера вам необходимо создать локальную копию образца статического сайта в формате HTML, а затем отправить файлы в свое веб-приложение Azure. Этот рабочий процесс показан на рисунке 3.4.

Рис. 3.4. Вы создаете локальную копию образцов файлов из GitHub с помощью команды `git clone`. Чтобы отправить эти локальные файлы в веб-приложение Azure, воспользуйтесь командой `git push`.

Попробуйте сейчас

Чтобы получить копию образца страницы в формате HTML из GitHub и отправить ее в веб-приложение, выполните приведенные ниже действия:

1 Откройте Cloud Shell на портале Azure и подождите несколько секунд до подключения к сеансу. Чтобы начать работу, вам нужен пример HTML-сайта из GitHub.

Чтобы *клонировать*или скопировать образец сайта в формате HTML из GitHub, введите команду:

```
git clone https://github.com/foouldsy/azure-mol-samples-2nd-ed.git
```

Если вы впервые используете Git в Cloud Shell, вам нужно определить несколько параметров в Git для идентификации. Для большинства упражнений в книге это не имеет значения, но при использовании с собственными проектами и приложениями это отличный способ отслеживать, кто выполняет определенные действия в системе управления версиями. Эти параметры задаются только раз. Введите свой адрес электронной почты и полное имя в `git config` следующим образом:

```
git config --global user.email "iain@azuremol.com"
git config --global user.name "Iain Foulds"
```

2 Измените каталог azure-mol-samples-2nd-ed, созданный при клонировании репозитория Git:

```
cd azure-mol-samples-2nd-ed/03/prod
```

3 Чтобы подготовиться к загрузке примера HTML-страницы, необходимо инициализировать Git, а затем добавить и зафиксировать свои файлы. Пока что не волнуйтесь о командах Git! Вам нужно сообщить Git, какие файлы следует отслеживать и добавлять, и оставить себе возможность отслеживать эти изменения позже:

```
git init && git add . && git commit -m "Pizza"
```

4 Теперь вы можете отправлять образец сайта в формате HTML в свое веб-приложение. Для начала определите учетные данные развертывания. Чтобы защитить веб-приложения при использовании такого метода развертывания, как Git или FTP, необходимо указать имя пользователя и пароль. Сервис «Веб-приложения» может использовать набор учетных данных, действительных для всех планов Azure App Service, или учетные данные уровня приложения, которые применяются только к одному приложению.

В реальном мире я рекомендую использовать учетные данные уровня приложения, чтобы свести к минимуму масштаб атаки, если одна из учетных данных будет скомпрометирована. Azure автоматически создает учетные данные уровня приложения, но их необходимо извлекать и назначать каждый раз. Чтобы все упростить, используйте определенные учетные данные, которые можно будет применять повторно в следующих главах.

Создайте учетные данные развертывания, а также введите имя пользователя и надежный пароль. Имя пользователя должно быть глобально уникальным. Если это поможет, добавьте свои инициалы к имени пользователя или воспользуйтесь правилами которое применяется в вашей среде.

```
az webapp deployment user set --user-name azuremol --password @azurem0l!
```

5 Затем нужно получить URL-адрес репозитория Git вашего веб-приложения. Введите имя веб-приложения (а не имя пользователя, созданное на шаге 4) и группу ресурсов, указанную при создании веб-приложения, для просмотра URL-адреса репозитория Git.

Как самостоятельно выполнить slash-dot

В следующем примере и последующих главах обратная косая черта (\) означает, что команда продолжается с новой строки. Это способ обработки длинных строк, используемый во многих онлайн-примерах, где вы копируете команды для вставки. В примерах в этой книге необязательно вводить символ обратной косой черты! Продолжайте набирать дополнительные параметры в одну длинную строку.

Если вы используете командную строку Windows, а не оболочку Bash, не используйте знаки обратной косой черты. Иначе вы точно не добьетесь желаемого результата!

```
az webapp deployment source config-local-git \
--name azuremol \
--resource-group azuremolchapter3 -o tsv
```

6 Ваше веб-приложение настроено для работы с репозиториями Git, поэтому вам нужно сообщить о репозитории в Cloud Shell. В Git вы определяете эти расположения как *удаленные*.

Скопируйте URL-адрес клона Git из шага 5, а затем задайте этот адрес как назначение для образца сайта в формате HTML в Cloud Shell с помощью указанной команды:

```
git remote add azure your-git-clone-url
```

7 Чтобы загрузить или скопировать файлы с помощью Git, вы выполняете команду *push*. Куда Git отправляет их? В *удаленное* расположение — как то, что вы настроили на шаге 6, например *Azure*. Последняя часть команды — это ветвь (обычно *master*). Ветвление в Git — это ваш способ отслеживать различные модели выполняемых работ. В производственных средах рекомендуется отправлять содержимое в ветви выпусков, которые можно называть как угодно, например *dev* или *staging*. Эти дополнительные ветви позволяют рабочему коду функционировать как обычно. Затем вы можете безопасно и без последствий для реальных рабочих нагрузок клиентов работать с новыми функциями или с обновлениями.

Отправьте образец сайта в формате HTML в свое веб-приложение:

```
git push azure master
```

8 При запросе введите пароль, созданный вами для учетных данных развертывания. Вы можете скопировать и вставить пароль, чтобы свести вероятность ошибки к минимуму.

В выводе показано, что у веб-приложения удалена страница стандартного сайта, а образец сайта в формате HTML загружен и настроен для запуска. Пример выходных данных:

```
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 510 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: Updating branch 'master'. remote: Updating submodules.
remote: Preparing deployment for commit id 'dda01e9d86'.
remote: Generating deployment script.
remote: Generating deployment script for Web Site
remote: Running deployment command...
remote: Handling Basic Web Site deployment.
remote: Creating app_offline.htm
remote: KuduSync.NET from: 'D:\home\site\repository' to:
     'D:\home\site\wwwroot'
remote: Deleting file: 'hostingstart.html'
remote: Copying file: 'index.html'
remote: Deleting app_offline.htm
remote: Finished successfully.
remote: Running post deployment command(s)...
remote: Deployment successful.
To https://azuremolikf@azuremol.scm.azurewebsites.net/azuremol.git
  * [new branch]       master -> master
```

Чтобы просмотреть обновленное веб-приложение, обновите сайт в браузере или откройте его повторно в окне «Обзор» на портале Azure. Он должен выглядеть как замечательный пример на рисунке 3.5. Да, сайт простой. Но и типовой статический сайт в формате HTML для веб-приложения, написанного на сложном языке .NET или Node. js, развертывается точно так же!

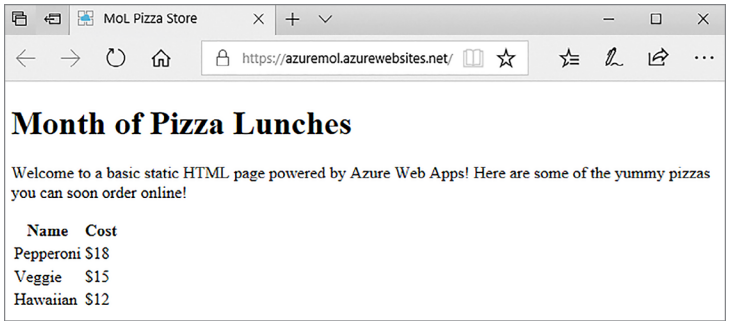


Рис. 3.5. Обновите страницу в браузере и вместо стандартной страницы веб-приложения вы увидите простой статический сайт в формате HTML из GitHub.

3.3 *Просмотр журналов диагностики*

Мы создали простое веб-приложение и развернули для него простой сайт в формате HTML, но что насчет общего управления? В случае проблем полезно заглянуть в журналы веб-сервера и приложений. Вы можете записать выходные данные из приложения в файлы журнала. Это поможет устранить неисправность. Файлы журналов можно просматривать в режиме реального времени или записывать для просмотра позже.

Ваше веб-приложение в основном работает самостоятельно. Вы почти не можете обслуживать его на уровне веб-узла. Если приложение работает с ошибками, скорее всего, вы захотите просмотреть журналы, чтобы узнать причины и устранить неполадки. С помощью сервиса «Веб-приложения Azure» можно настраивать такие параметры, как уровень просматриваемых сообщений журнала, а также расположение и срок хранения журналов. На рисунке 3.6 показано, как создавать и просматривать файлы журналов с помощью сервиса «Веб-приложения».

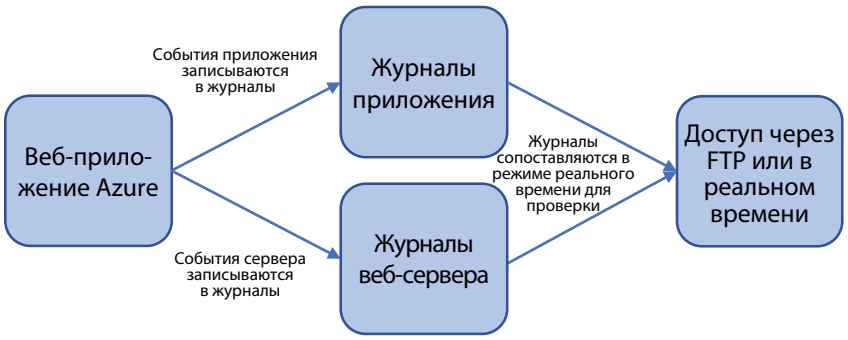


Рис. 3.6. Ваше приложение может создавать журналы приложения и журналы сервера. Вы можете скачать эти журналы через FTP или просмотреть их в режиме реального времени для анализа и решения проблем.

Попробуйте сейчас

Чтобы настроить веб-приложение для журналов диагностики, выполните приведенные ниже действия:

1. На портале Azure выберите созданное вами в предыдущем упражнении веб-приложение.
2. В окне «Обзор» прокрутите вниз до раздела «Мониторинг» и выберите пункт «Журналы App Service».
3. Просмотрите доступные параметры журнала, например детализацию и состояние трассировки неудачных запросов. Если вы работаете на уровне инфраструктуры Azure, вероятно, вы вместе с разработчиками приложений определите нужные им журналы. Затем можно включить ведение соответствующих журналов. Журналы могут храниться в локальной файловой системе веб-приложения или отправляться в хранилище Azure Storage для обработки с помощью другого приложения.
4. Пока же включите параметр «Ведение журнала приложений (файловая система)». Кроме того, включите ведение журнала веб-сервера в файловой системе со сроком хранения 7 дней. Уровень ошибки по умолчанию может ничего не показывать, если все работает правильно, но будьте осторожны с переходом на уровень «Отладка» или «Трассировка», так как журналы могут быстро переполниться, что существенно усложнит понимание ситуации! Если возникла проблема, постепенно увеличивайте уровень журнала, пока не соберете достаточно сведений для устранения неполадок, не перегружая журнал данными.

Если вы действительно хотите подробно погрузиться в данные, вы можете изучить журналы, хранящиеся в файловой системе, по протоколу FTP. FTP-адреса отображаются в разделе «Скачать журналы» или в окне «Обзор» веб-приложения. Вы можете подумать: «FTP — сложный способ получить журналы диагностики. Разве нет ничего попроще?» Конечно же есть! На портале Azure — там, где вы настраивали журналы диагностики, — указан параметр «Поток журналов». Угадайте, что он делает? Я подскажу. Это связано с потоковой передачей ваших файлов журналов.

Нажав эту кнопку на портале Azure, можно выбрать «Журналы приложений» или «Журналы веб-сервера». Эти журналы считываются из тех же журналов диагностики, которые записываются в файл. Данные могут появиться в потоке через несколько минут, а отображение зависит от заданных вами уровней журналов и от того, создает ли ваше веб-приложение какие-либо события. Для простого сайта в формате HTML поток довольно скучен, но это отличная функция для браузера. На рисунке 3.7 показан пример потоковой передачи журналов веб-сервера на портале Azure.

Попробуйте сейчас

Просмотрите файлы журнала потоковой передачи на портале Azure. Вам, вероятно, потребуется обновить страницу в веб-браузере несколько раз, чтобы создать события в журналах.

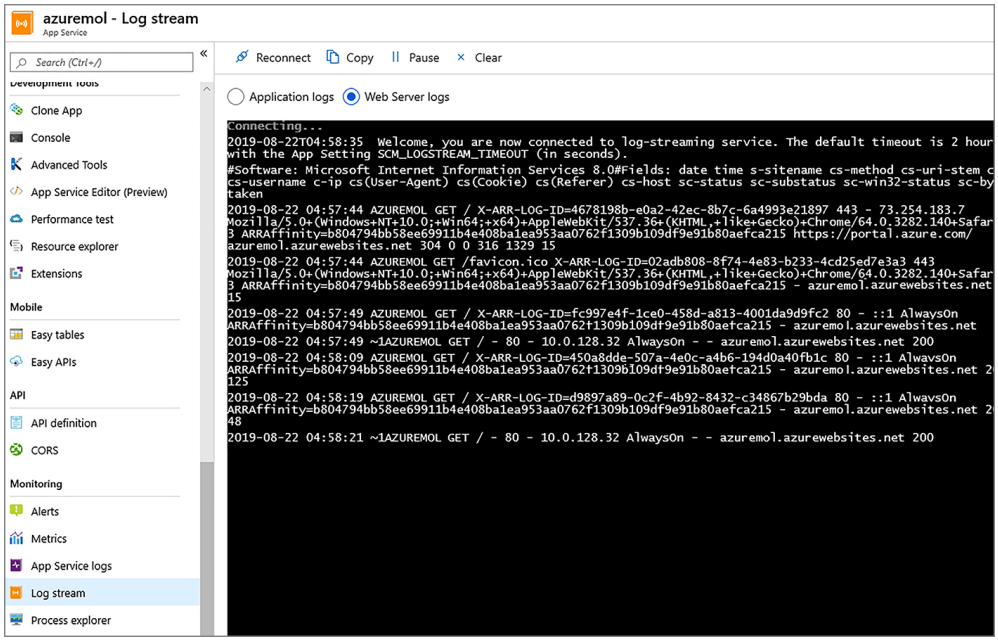


Рис. 3.7. Чтобы проверить и отладить производительность приложения, можно просматривать потоки журналов веб-сервера для сервиса «Веб-приложения» в режиме реального времени прямо в своем приложении. В окне консоли в правой части экрана отображаются журналы потоковой передачи в реальном времени из вашего веб-приложения.

Изучая Azure и используя интерфейс командной строки Azure или модуль Azure PowerShell, вы научитесь передавать журналы с помощью этих средств. Разработчики также могут включить удаленную отладку с помощью Visual Studio или настроить Application Insights, чтобы разрешить веб-приложению предоставлять данные телеметрии дополнительным сервисам мониторинга и диагностики. Ключевой вывод: при переходе к решениям PaaS, например к веб-приложениям, вы можете по-прежнему получать критически важные журналы диагностики и данные приложений, чтобы устранять неполадки и контролировать производительность веб-приложения.

3.4 *Лабораторная работа: создание и использование слота развертывания*

Мы рассмотрели создание простого сайта в формате HTML и отправку страницы в сервис «Веб-приложения Azure» с помощью Git. Но как добавить на сайт новые виды пиццы и просмотреть их перед публикацией и заказами от клиентов? Теперь мы узнаем, как использовать слот развертывания для загрузки изменений, их проверки и переключения в производственную среду:

1. В своем веб-приложении выберите «Слоты развертывания». Добавьте слот развертывания с именем Dev, но не клонируйте параметры из существующего слота развертывания.

2 Когда все будет готово, выберите из списка промежуточный слот. На портале отображаются те же параметры конфигурации и ведения журнала, что и в производственном слоте. Это позволяет понять, как можно изменить параметры в этом слоте развертывания, не влияя на действующий сайт.

3 На этот раз изучите параметры на портале Azure для центра развертывания. Вам нужно использовать Local Git в качестве системы управления версиями, которая использует сервис построения App Service для промежуточного слота. Это произошло в фоновом режиме, когда вы использовали Azure CLI в предыдущем упражнении, но существуют и другие варианты источников для развертывания кода и сервиса, который выполняет сборку и создает развертывание.

4 После завершения скопируйте URI клона Git, например https://azuremol-dev.scm.azurewebsites.net:443/azuremol.git. Обратите внимание, что репозиторий Git включает в себя `-dev` для промежуточного слота.

Пример сайта разработки включен в образцы GitHub, клонированные вами ранее.

5 В Azure Cloud Shell перейдите в каталог разработки следующим образом:

```
cd ~/azure-mol-samples-2nd-ed/03/dev
```

6 Как и прежде, инициализируйте, добавьте и зафиксируйте свои изменения в Git с помощью таких команд:

```
git init && git add . && git commit -m "Pizza"
```

7 Снова создайте ссылку на новый репозиторий Git в промежуточном слоте с помощью команды `git remote add dev` и укажите после нее URL-адрес развертывания Git своего промежуточного слота.

8 Используйте команду `git push dev master`, чтобы отправить изменения в слот развертывания.

9 Выберите URL-адрес своего промежуточного слота в окне «Обзор» на портале Azure. Конечно, изменение не очень существенное, но заголовок страницы позволяет вам узнать, что вы видите версию для разработки.

10 Как вы думаете, что произойдет, если нажать кнопку «Поменять» на портале Azure для веб-приложения, как показано на рисунке 3.8? Попробуйте, а затем обновите главную страницу, например https://azure-mol.azurewebsites.net, в браузере.

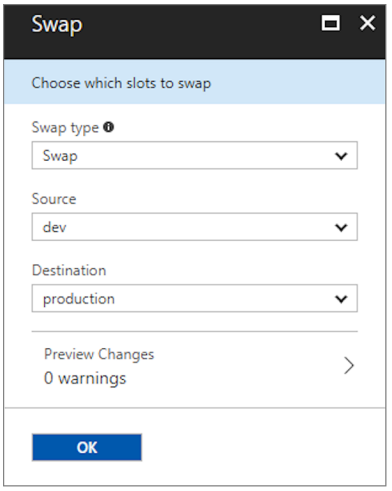


Рис. 3.8 При замене слотов для веб-приложения вы выбираете исходный и конечный экземпляры для изменения. Вы также можете узнать, как все будет выглядеть, прежде чем применить изменения.

Слоты развертывания: за кулисами

После переключения слотов содержимое *производственного* слота окажется в слоте *dev*, а контент *dev* переместится в *производственную среду*. Не все параметры можно переключать, например это не сработает с параметрами SSL и личными доменами. Но в целом слоты развертывания — это отличный способ размещать и проверять содержимое перед тем, как его увидят ваши клиенты. Кроме того, вы можете выполнить переключение с *предварительным просмотром*. Таким образом вы убедитесь, что переключаемое содержимое работает правильно, прежде чем задействовать его в производственной среде.

Вы также можете настроить автоматическое переключение для рабочих процессов DevOps. В этом случае при фиксации кода в системе управления версиями, например в GitHub, можно активировать сборку в слоте развертывания сервиса «Веб-приложения Azure». После того как сборка завершена и приложение готово к передаче содержимого, слоты развертывания автоматически переключаются, чтобы задействовать код в производственной среде. Как правило, этот рабочий процесс используется в тестовой среде, чтобы просматривать изменения кода перед публикацией в производственной среде!

4 Знакомство с сервисом хранилища Azure

В мире ИТ можно быть уверенным в одном: когда все плохо, виноваты сети и хранилища. Говорю это как бывший администратор сети SAN! Я был главным фанатом сетевых технологий. Я, конечно, шучу (насчет фаната), но вообще-то неважно, насколько хорошо разработано и написано приложение, если не работают базовые элементы поддерживающей его инфраструктуры. Наберитесь терпения, пока мы будем изучать сервисы хранилища и сетей Azure в паре следующих глав. У вас может быть искушение пропустить эти сервисы, чтобы сразу перейти к самому интересному, но время на изучение этих базовых моментов окупится с лихвой. Они, конечно, не сделают вашу еду вкуснее, но помогут клиентам заказать доставку вкусной пиццы!

В этой главе описываются различные типы хранилища в Azure и сценарии их использования. Я также расскажу о вариантах избыточности и репликации для сервиса хранилища Azure, а также о том, как обеспечить максимальную производительность ваших приложений.

4.1 Управляемые диски

Много лет назад серверные хранилища были дорогостоящими, медленными и слишком сложными. Зачастую поставщики хранилищ продавали оборудование за сотни тысяч долларов, а затем армия консультантов и инженеров настраивали его много дней или даже недель. Когда виртуализация поселилась в ЦОД, а VMware и Hyper-V стали доступнее, начались проблемы с хранилищами. Не говоря уже о несоответствии микропрограмм в адаптерах серверных хранилищ и в массивах хранения данных, о постоянных сбоях избыточных сетевых путей и твердотельных дисках (SSD), считавшихся единственным способом повысить производительность.

Так что же, облако Azure волшебным образом решило все эти проблемы? Конечно нет! Но оно обрабатывает 95 % этих проблем и позволяет вам сконцентрироваться на проектировании и создании великолепных интерфейсов для клиентов. В этой главе рассматриваются последние 5 %, о которых вам следует знать.

Сервис управляемых дисков Azure упрощает реализацию хранилища виртуальных машин. Управляемые диски абстрагируют большую часть фоновой работы, предоставляя вам... . . диск. Это все, о чем вам действительно нужно заботиться для Вм: насколько они велики, производительны и к чему они подключаются. На протяжении всей книги и во всех реальных развертываниях всегда следует использовать управляемые диски для ВМ. Управляемые диски — это вариант по умолчанию, и нет многих веских причин что-то менять.

До этого приходилось создавать учетную запись хранения с уникальным именем, ограничивать число создаваемых дисков в каждой из них и вручную перемещаться по пользовательским образам дисков, чтобы создать ВМ в других регионах. Такие типы дисков называют *неуправляемыми* или *классическими*. Сервис «Управляемые диски» не требует создавать учетную запись хранилища, предоставляет 10 000 дисков на подписку и позволяет создавать ВМ на основе пользовательских образов дисков во всех регионах. Кроме того, появилась возможность создавать и использовать снимки дисков, автоматически шифровать незадействованные данные и использовать диски объемом до 64 ТиБ.

Почему это важно? Вы можете наткнуться на старую документацию или публикации в блогах, где рекомендуется создавать учетную запись хранения для ВМ. Остановитесь! Да, вы можете преобразовать ВМ из неуправляемых дисков в управляемые, но если вы начинаете с чистого листа, лучше сразу создавать управляемые. Неуправляемые диски используются скорее для обратной совместимости с существующими разработками, хотя я бы поспорил, нужно ли преобразовывать эти рабочие нагрузки в управляемые диски.

4.1.1 Диски ОС

Помните, я говорил, что для наилучшей производительности требовалось покупать диски SSD? Извините, но волшебной таблетки от этого в Azure нет. Простите. Правда в том, что SSD значительно быстрее обычных вращающихся дисков. У последних физически ограничена скорость... . . Правильно. Вращения. Инженеры Microsoft пока не смогли победить законы физики! Обычные вращающиеся диски все еще используются как недорогое архивное хранилище, и современные технологии обеспечивают неплохую производительность на основе пула таких дисков — как и в обычных массивах хранения.

Первое и главное решение при выборе ВМ Azure — это тип используемого хранилища:

- *Диски SSD категории «Премиум»* — используйте скоростные SSD-диски для оптимальной производительности, увеличения числа операций ввода-вывода в секунду и низкой задержки. Это рекомендуемый тип хранилища для большинства рабочих нагрузок.
- *Твердотельные накопители категории «Стандарт»* — используйте стандартные SSD-диски для стабильной производительности на уровне жестких дисков. Эти диски отлично подойдут для разработки и тестовых рабочих нагрузок, а также для дешевых и нетребовательных производственных сред.
- *Диски HDD категории «Стандарт»* — это обычные вращающиеся диски для эпизодического доступа к данным, например к архивам или резервным копиям.

Необходимый тип хранилища определяется, исходя из размера ВМ. В главе 2 при создании ВМ мы выбрали размер, обеспечивший быстрое развертывание. По умолчанию, вероятно, использовалась ВМ серии D2S_v3, которая предоставляла доступ к дискам SSD категории «Премиум». Как же определить, у каких ВМ есть доступ к дискам SSD категории «Премиум»? Ищите букву *s* в размере ВМ, обозначающую *SSD*. Есть несколько исключений из правила,

но это полезный шаблон. Следующие примеры помогут определить, какие виртуальные машины могут получать доступ к дискам категории «Премиум», а какие — к стандартным SSD- или жестким дискам.

- Диски SSD категории «Премиум» доступны для ВМ серий D2S_v3, F*s*, GS, и L*s*.
- Виртуальные машины серий D, A, F и M ограничены стандартными SSD- или HDD-дисками.

Выбрав размер ВМ, совместимый с дисками SSD категории «Премиум», вы не обязаны использовать именно их. Вы можете создавать и использовать стандартные диски SSD или HDD. Выбирая диски SSD категории «Премиум», вы планируете будущее своего приложения. При необходимости вы сможете пользоваться высокопроизводительными дисками SSD, не меняя размера ВМ и без простоев. Но ВМ любого размера может использовать стандартные диски SSD.

Эфемерный диск ОС

Существует особый тип диска ОС, которые называют *эфемерным диском*. Это все еще управляемый диск своего рода, но он является локальным для базового хоста Azure. Из-за этого эфемерный диск очень быстрый и с низкой задержкой.

Так как данные не записываются в удаленный массив хранилища, они могут не сохраняться после перезагрузки ВМ, если вы переходите на другой базовый хост. Эфемерные диски прекрасно подходят для рабочих нагрузок без сохранения состояния, они могут обрабатывать загрузку с чистым образом каждый раз, и им не требуется хранить данные локально для сохранения доступа после перезагрузки.

Только ВМ определенных размеров поддерживают эфемерные диски, но нет никаких дополнительных затрат на их использование. Кроме того, они доступны во всех регионах. Вы теряете некоторые функциональные возможности, такие как Azure Site Recovery и шифрование дисков Azure (главы 13 и 14 соответственно), однако если вам необходимо высокоскоростное хранилище с низкой задержкой, обратите внимание на эфемерные диски.

Попробуйте сейчас

Как определить доступные вам размеры ВМ? На портале Azure откройте Cloud Shell. Введите такую команду (можете использовать свой регион):

```
az vm list-sizes --location eastus --output table
```

Запомните, что любой размер с буквой s дает доступ к дискам SSD категории «Премиум».

4.1.2 Временные диски и диски с данными

Вы поняли, какой уровень производительности нужен для ваших приложений, теперь рассмотрим еще пару фрагментов головоломки. Диски подключаются 2 способами:

- *Временные диски* — ко всем ВМ автоматически подключается локальное хранилище SSD из небольшого хранилища высокой производительности на базовом узле. Будьте внимательны при использовании этого временного диска! Как следует из названия, этот диск подключен к ВМ временно. Если ВМ перемещается на новый хост во время обслуживания, будет подключен новый временный диск и все данные, которые на нем хранятся, будут потеряны. Временный диск используется как рабочее пространство или кэш приложения.

- *Диски с данными* — все диски, созданные и подключенные к конкретной ВМ, действуют ожидаемо с точки зрения разделов, файловых систем и постоянных точек подключения. Когда ВМ перемещается по ЦОД Azure, диски с данными подключаются повторно и находятся там, где хранится основная часть приложений и данных. Вы по-прежнему можете использовать дисковые пространства или программный механизм RAID для пула дисков с данными на уровне ВМ, чтобы еще больше повысить производительность.

Существует определенный тип диска данных, который можно подключить к ВМ, если вам требуется максимальная производительность и низкая задержка. Это диски Ultra. Эти диски на уровень выше SSD-дисков категории «Премиум» и доступны только для дисков с данными. Диски Ultra предназначены для крупных баз данных и рабочих нагрузок с интенсивным использованием данных, таких как SAP HANA. О какой скорости мы говорим? На момент написания этой книги диски Ultra поддерживали размер до 64 ТиБ и скорость до 160 000 операций ввода-вывода в секунду на диск с максимальной пропускной способностью 2000 Мбит/с.

4.1.3 Параметры кэширования дисков

Кроме того, важно учитывать диск ОС, поставляемый вместе с ВМ. При создании ВМ вы всегда получаете хотя бы один диск, на котором установлена сама ОС. Может возникнуть желание использовать его для установки приложений или записи на него файлов журнала. Но с системного диска лучше вообще не запускать приложений, за исключением небольших развертываний для подтверждения концепции! Иначе, скорее всего, вы не получите требуемую производительность.

Для дисков в Azure можно задать политику кэширования. По умолчанию на диске ОС применяется кэширование для *чтения и записи*. Этот тип кэширования не оптимален для рабочих нагрузок приложений, записывающих, например файлы журналов или базы данных. Для дисков с данными, напротив, задана стандартная политика: *не кэшировать*. Это хороший выбор для рабочих нагрузок, выполняющих множество операций записи. Кроме того, вы можете применить политику кэширования *только для чтения*. Она подходит для рабочих нагрузок приложений, которые в основном читают данные с дисков.

В целом всегда подключайте и используйте диски с данными для установки и запуска приложений. Даже стандартная политика «не кэшировать», скорее всего, обеспечит лучшую производительность, чем политика кэширования «чтение и запись» на диске ОС.

4.2 Добавление дисков к ВМ

Из этого раздела вы узнаете, как добавлять диски на ВМ при ее создании. В главе 2 мы создали ВМ с помощью портала Azure. Теперь создадим ВМ посредством Azure CLI. Azure CLI позволяет быстро создать ВМ и сразу же подключить диск с данными.

Попробуйте сейчас

Чтобы создать ВМ и увидеть диски с данными в действии, выполните приведенные ниже действия:

1. В Azure Cloud Shell создайте группу ресурсов с помощью команды `az group create`, введя имя группы ресурсов вместе с расположением:

```
az group create --name azuremolchapter4 --location eastus
```

2. Создайте ВМ с помощью команды `az vm create`. Последний параметр, `--data-disk-sizes-gb`, позволяет создать диск с данными вместе с ВМ. В практическом задании в конце раздела вы сможете подключиться к этой ВМ и инициализировать диски.
   - Для этого упражнения вы можете создать ВМ Linux или ВМ Windows. Если вы знакомы с Linux или хотите узнать, как инициализировать и подготовить диск для Linux, создайте ВМ Ubuntu LTS с помощью следующей команды:

     ```
     az vm create \
       --resource-group azuremolchapter4 \
       --name storagevm \
       --image UbuntuLTS \
       --size Standard_B1ms \
       --admin-username azuremol \
       --generate-ssh-keys \
       --data-disk-sizes-gb 64
     ```

   - Если вам удобнее работать в Windows, используйте эту команду для создания ВМ Windows Server 2019. Затем вы сможете подключиться к ВМ через RDP для настройки дисков:

     ```
     az vm create \
       --resource-group azuremolchapter4 \
       --name storagevm \
       --image Win2019Datacenter \
       --size Standard_B1ms \
       --admin-username azuremol \
       --admin-password P@ssw0rd! \
       --data-disk-sizes-gb 64
     ```

   - Создание ВМ займет несколько минут. У ВМ уже есть один диск с данными, подключенный и готовый к использованию.

Как добавить диск с данными через несколько недель или месяцев? Снова используем Azure CLI, чтобы увидеть, как быстро добавить новый диск. Процесс одинаков для ВМ Linux и Windows. Просто укажите Azure создать новый диск и присоединить его к ВМ.

Попробуйте сейчас

Добавьте дополнительный диск с данными к ВМ, как показано далее.

Создайте новый диск с помощью команды `az vm disk attach`. Укажите его имя и размер. Помните предыдущее обсуждение дисков категорий «Стандарт» и «Премиум»? В следующем примере мы создаем диск SSD категории «Премиум»:

```
az vm disk attach \
    --resource-group azuremolchapter4 \
    --vm-name storagevm \
    --name datadisk \
    --size-gb 64 \
    --sku Premium_LRS \
    --new
```

Вы узнаете последнюю часть этого типа хранения? *LRS* означает *локально избыточное хранилище*. Мы рассмотрим варианты избыточности в разделе 4.3.3.

Используя всего 2 команды в Azure CLI, вы создали ВМ, включая диск с данными, а затем смоделировали подключение дополнительного диска. Просто подключив эти диски, вы не сможете сразу записывать на них данные. Любой диск — физический, подключенный к локальному серверу, или виртуальный, присоединенный к ВМ, — необходимо инициализировать, создать на нем разделы и файловую систему. Вы можете сделать это в дополнительном упражнении в конце главы.

4.3 Хранилище Azure

Может показаться, что хранилища необязательны для создания и запуска приложений, но, на самом деле, это комплексный сервис, охватывающий гораздо больше, чем вы представляете. Сервис хранилища Azure предлагает не просто место для хранения файлов или виртуальных дисков для ваших ВМ.

Давайте посмотрим, что требуется вашей вымышленной компании по доставке пиццы, чтобы создать приложение, обрабатывающее заказы клиентов. Приложению нужно хранилище данных о доступных видах пиццы, списке начинок и ценах. По мере получения и обработки заказов понадобится способ обмена сообщениями между различными компонентами приложения. На веб-сайте необходимо разместить аппетитные изображения пиццы для клиентов. Как видно на рисунке 4.1, сервис хранилища Azure предоставляет множество функций хранения и может удовлетворить все 3 из этих потребностей.

Рис. 4.1. Учетная запись сервиса хранилища Azure позволяет создавать и использовать разнообразные функции хранения, намного превосходящие простое хранение файлов.

- *Хранилище BLOB-объектов* — для неструктурированных данных, например мультимедийных файлов и документов. Приложения могут хранить данные, к примеру изображения, в хранилище BLOB-объектов, а затем обрабатывать их. В хранилище BLOB-объектов вы хранили бы изображения пиццы.
- *Хранилище таблиц* — предназначено для неструктурированных данных в хранилище NoSQL. Выбирая между хранилищами данных SQL и NoSQL, планируйте приложение и оценивайте требования к производительности для обработки больших объемов данных. Хранилище таблиц подошло бы для перечней видов пиццы в вашем меню. В разделе 4.3.1 NoSQL рассматривается подробнее.

- *Хранилище очередей* — предназначено для взаимодействия между различными уровнями и компонентами облачных приложений. В нем создаются, считываются и удаляются сообщения, которыми обмениваются компоненты приложения. Вы могли бы использовать хранилище очередей, чтобы передавать сообщения между веб-интерфейсом, где клиент делает заказ, и внутренним сервисом, где вы обрабатываете заказы и выпекаете пиццу.
- *Хранилище файлов* — используется для старого доброго SMB: общего файлового ресурса, доступного как для Windows, так и для платформ Linux и macOS. Оно часто применяется для сбора журналов от виртуальных машин.

Хранилища Azure для ВМ вполне понятны. Вы создаете и используете управляемые диски Azure — тип виртуального жесткого диска (VHD), решающего многие вопросы в части производительности и распространения виртуальных дисков через платформу. Вы создаете ВМ, присоединяете любые управляемые диски с данными и позволяете платформе Azure определять избыточность и доступность.

4.3.1 Хранилище таблиц

Давайте рассмотрим несколько различных типов хранилищ данных. Во-первых, *хранилище таблиц*. Вероятно, многие уже знакомы с традиционной базой данных SQL, например с Microsoft SQL Server, MySQL или PostgreSQL. Это *реляционные базы данных*, состоящие из одной или нескольких таблиц с одной или несколькими строками данных. Реляционные базы данных используются для разработки приложений и могут структурировано проектироваться, визуализироваться, а также позволяют выполнять структурированные запросы — буква *S* в *SQL* (т. е. язык структурированных запросов).

Базы данных NoSQL немного отличаются. Они не следуют единому структурированному подходу, и данные не хранятся в таблицах, каждая строка которых содержит одинаковые поля. Существуют различные реализации баз данных NoSQL, например MongoDB и CouchDB. Базы данных NoSQL хвалят за горизонтальную масштабируемость (вы добавляете серверы вместо памяти или ЦП) и эффективную обработку больших наборов данных.

Способы хранения данных в базе данных NoSQL распределяются по нескольким категориям.

- *Ключ-значение*, например Redis
- *Столбец*, например Cassandra
- *Документ*, например MongoDB

Каждый подход имеет преимущества и недостатки с точки зрения производительности, гибкости или сложности. В таблице хранилища Azure хранятся пары «ключ-значение», что позволяет получить начальные знания о базах данных NoSQL, если ранее вы использовали базы данных SQL, например Microsoft SQL или MySQL.

Если вам нравится визуализировать данные, загрузите и установите «Обозреватель сервиса хранилища Azure» с сайта https://azure.microsoft.com/features/storage-explorer. Необязательно делать это прямо сейчас. «Обозреватель хранилищ» — это отличный инструмент, чтобы познакомиться с работой таблиц и очередей. В этой главе мы не будем детально изучать базы данных NoSQL, но в главе 10 подробнее остановимся на интересных и полезных базах данных NoSQL с помощью Azure Cosmos DB. По сути, в следующем упражнении вы будете использовать API Cosmos DB, чтобы подключиться к хранилищу Azure и создать в нем таблицу. Использование таблиц Azure — это, скорее, введение в базы данных NoSQL, чем серьезный пример для производственной среды.

А теперь давайте запустим небольшое приложение, чтобы увидеть, как добавлять данные в рабочую программу и запрашивать их оттуда. Эти простые примеры показывают, как хранятся различные виды пиццы и сколько они стоят. Мы не будем использовать большие базы данных типа Microsoft SQL Server или MySQL, а применим базу данных NoSQL с хранилищем таблиц Azure.

Попробуйте сейчас

Чтобы увидеть таблицы Azure в действии, выполните приведенные ниже действия:

1. Зайдите на портал Azure в браузере и откройте Cloud Shell.
2. В главе 3 вы получили копию примеров для Azure из GitHub. Если нет, копию можно получить так:

   ```
   git clone https://github.com/foouldsy/azure-mol-samples-2nd-ed.git
   ```

3. Зайдите в каталог с примерами для сервиса хранилища Azure.

   ```
   cd ~/azure-mol-samples-2nd-ed/04
   ```

4. Установите несколько зависимостей Python, если их еще нет. Теперь установите пакет `azurerm`, позволяющий создавать и контролировать ресурсы Azure, а также 2 пакета `azure` (базовые пакеты SDK для Python в Azure Cosmos DB и в хранилище):

   ```
   pip install --user azurerm azure-cosmosdb-table azure-storage-
   ➥queue==2.1.0
   ```

   Что означает `--user` при установке пакетов? В Azure Cloud Shell невозможно установить пакеты в основную систему. У вас нет разрешений. Поэтому пакеты устанавливаются в среде пользователя. Причем в разных сеансах, что позволяет использовать в этих примерах полностью чистые пакеты SDK для Azure.

5. Запустите пример приложения Python для таблиц. Следуйте подсказкам, чтобы насладиться пиццей:

   ```
   python storage_table_demo.py
   ```

Змеиный полет

Python — популярный язык программирования, часто изучаемый на курсе «Введение в компьютерные науки». Python пригодится в ИТ-операциях или администрировании. Это мощный и совместимый с разными ОС язык для написания скриптов. На Python можно писать не только скрипты, но и создавать сложные приложения. Например, вы использовали Azure CLI, написанный на Python.

Я использую Python в некоторых примерах из этой книги, чтобы они работали вне Cloud Shell без каких-либо изменений. Python встроен в дистрибутивы macOS и Linux. Пользователи Windows могут загрузить и быстро установить Python и запускать эти скрипты локально. Python отлично подойдет для пользователей без опыта программирования, а также для опытных разработчиков, знакомых с другими

языками. Документация Azure для сервиса хранилища Azure и многих других сервисов поддерживает различные языки, в том числе .NET, Java и Node. js. Вы можете создавать приложения с таблицами и на других языках.

Третье издание книги *The Quick Python Book* Наоми Седер (http://mng.bz/6QZA) поможет вам ускорить работу, если вы хотите узнать больше. Кроме того, есть видеокурс *Get Programming with Python in Motion*, созданный Аной Белл (Ana Bell). Он доступен по адресу http://mng.bz/oPap.

4.3.2 Хранилище очередей

Таблицы Azure интересны и полезны, когда вы только начинаете изучать разработку облачных приложений. Начиная создавать и контролировать приложения в облаке, вы, как правило, разбиваете их на мелкие компоненты, каждый из которых самостоятельно масштабирует и обрабатывает данные. Чтобы разрешить этим компонентам взаимодействовать и обмениваться данными, обычно используется определенная очередь сообщений. Займите очередь в Azure.

Очереди Azure позволяют создавать, считывать и удалять сообщения с небольшими фрагментами данных. Такие сообщения создаются и извлекаются различными компонентами приложения при обмене данными. Очереди Azure не удаляют сообщение, пока приложение не завершит обработку его данных.

Попробуйте сейчас

Чтобы увидеть очереди Azure в действии, запустите этот сценарий Python в том же каталоге azure-samples/4. Следуйте инструкциям, чтобы просмотреть сообщения, записанные в очередь, прочитанные в ней и удаленные оттуда:

```
python storage_queue_demo.py
```

Перейдем к примеру приложения, обрабатывающего заказы пиццы. У вас может быть интерфейсный компонент приложения, в котором клиенты заказывают пиццу, и очередь сообщений, которые передаются серверному компоненту приложения, где вы обрабатываете полученные заказы. После получения заказов сообщения могут выглядеть, как показано на рисунке 4.2.

ID	Message Text	Insertion Time (UTC)	Expiration Time (UTC)	Dequeue Count	Size
ca57a12c-21b8-4640-9e07-4fc3a81c8dd5	Veggie pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	21 B
d68f90a9-1d5a-4a0e-af79-f285efa2aca2	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B
7f3c6f4a-9d47-488f-9344-1cb5bbec0fa4	Hawiian pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	22 B
63f07e06-ab0d-48c0-81c9-019d4255f335	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B
66b48f73-d136-4d82-9b41-f32f93f3d725	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B

Рис. 4.2. Сообщения от интерфейсного компонента со сведениями о пицце, заказанной каждым клиентом, в свойстве Текст сообщения.

По мере того как серверный компонент приложения обрабатывает каждый заказ на пиццу, сообщения удаляются из очереди. На рисунке 4.3 показано, как удаляется первое сообщение очереди, после того как вегетарианская пицца отправлена в духовку.

ID	Message Text	Insertion Time (UTC)	Expiration Time (UTC)	Dequeue Count	Size
d68f90a9-1d5a-4a0e-af79-f285efa2aca2	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B
7f3c6f4a-9d47-488f-9344-1cb5bbec0fa4	Hawiian pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	22 B
63f07e06-ab0d-48c0-81c9-019d4255f335	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B
66b48f73-d136-4d82-9b41-f32f93f3d725	Pepperoni pizza ordered.	Fri, 23 Aug 2019 03:39:39 GMT	Fri, 30 Aug 2019 03:39:39 GMT	0	24 B

Рис. 4.3. По мере обработки сообщения удаляются из очереди. Первое сообщение на рисунке 4.2 удалено после обработки серверным компонентом приложения.

4.3.3 Доступность и избыточность хранилища

Центры обработки данных Azure созданы отказоустойчивыми благодаря избыточности подключений к Интернету, генераторов питания, нескольких сетевых путей, массивов хранения и т. д. Однако вам все-таки придется выполнить свою работу при проектировании и запуске приложения. С помощью сервиса хранилища Azure вы выбираете требуемый уровень избыточности хранилища. Он зависит от приложения и критичности данных. Рассмотрим доступные варианты избыточности хранилища:

- *Локально избыточное хранилище (LRS)* — данные реплицируются трижды внутри одного ЦОД, в котором создана учетная запись хранения. Этот параметр обеспечивает избыточность в случае сбоя единичного оборудования, но если весь центр обработки данных выходит из строя (редко, но возможно), вы потеряете доступ к своим данным.
- *Хранилище, избыточное в пределах зоны (ZRS)*, — следующий уровень по сравнению с LRS. Здесь данные трижды реплицируются в двух или трех центрах обработки данных в регионе (с несколькими ЦОД) или в разных регионах. Хранилище ZRS также не ограничивается одной зоной доступности. Мы подробнее рассмотрим это в главе 7.
- *Геоизбыточное хранилище (GRS)* — в GRS, данные реплицируются трижды в основном регионе, в котором создано хранилище, а затем — трижды в связанном регионе. Связанный регион обычно находится на расстоянии сотен километров, если не больше. Например, западная часть США связана с восточной частью США, Северная Европа — с Западной Европой, а Юго-Восточная Азия — с Восточной Азией. GRS — это отличный вариант, чтобы обеспечить избыточность для рабочих приложений.
- *Геоизбыточное хранилище с доступом для чтения (RA-GRS)* — это вариант избыточности для данных категории «Премиум». Данные реплицируются в связанных регионах, как при GRS, но вы также получаете доступ для их чтения в этой дополнительной зоне.

4.4 Практическое задание: изучение сервиса хранилища Azure

А вот и шанс проверить свои навыки. Выберите одну из приведенных задач для практических упражнений.

4.4.1 Ориентация на ВМ

А чтобы войти в систему ВМ и убедиться, что инициализация диска и создание файловой системы такие же, как на любой известной вам ВМ, выполните одно из приведенных ниже упражнений.

1. Войдите в ВМ, созданную в разделе 4.2. Подключитесь с помощью SSH или через RDP по своему усмотрению.
2. Инициализируйте диск и создайте раздел.
   - В Linux последовательность такая: `fdisk`, `mkfs`, а затем `mount`.
   - В Windows используйте любую удобную последовательность, например «Управление дисками» > «Инициализировать» > «Создать том» > «Форматировать».

4.4.2 Ориентация на разработчика

Если вы разработчик и не хотите разбираться, как инициализировать диски с данными на ВМ, вернитесь в Cloud Shell и ознакомьтесь с двумя демонстрациями на языке Python, где используются таблицы и очереди. Даже новички в Python непременно все поймут:

- Продумайте сценарии для собственных приложений, где вы могли бы внедрить таблицы или очереди. Например, что нужно для создания изначально облачных приложений с отдельными компонентами, поддерживающими очереди?
- Измените один из примеров, чтобы добавить в меню еще одну пиццу (если это таблица) или создать новое сообщение о заказе (если это очередь).

5 Основные сведения о сервисе «Сети Azure»

В главе 4 мы изучили сервис хранилища Azure. К основным сервисам для облачных приложений также относится сервис «Сети Azure». В Azure есть множество мощных сетевых функций, которые глобально защищают и маршрутизируют трафик. Они помогают вам сосредоточиться на создании и обслуживании приложений и не волноваться об IP-адресах и таблицах маршрутизации. Если вы создаете и запускаете интернет-магазин для приема заказов на пиццу, он должен безопасно передавать данные о клиентах и обрабатывать платежи.

В этой главе мы исследуем виртуальные сети и подсети Azure, а также создание интерфейсов для них. Чтобы защитить и контролировать поток трафика, необходимо создать группы безопасности сети и правила. Если вы раньше не работали с сетями или прошло много времени с тех пор, как вы использовали IP-адреса и сетевые адаптеры, эта глава может отнять у вас немного больше времени. В ней представлено много упражнений «Попробовать». Однако вам определенно будет полезно уделить время на изучение этой главы, так как сети — это ключевой аспект многих сервисов в Azure.

5.1 Компоненты виртуальной сети

Вспомните, как много кабелей за вашим компьютерным столом. Теперь подумайте обо всех кабелях, необходимых для подключения компьютеров на данном этаже офисного здания. А теперь представьте, сколько их нужно, чтобы подключить компьютеры на одном этаже офисного здания. А если взять все здание? Вы бывали когда-нибудь в центре обработки данных или видели его на фото? Представляете, насколько большой ЦОД Azure! А как насчет десятков ЦОД по всему миру? Я не силен в математике, поэтому едва ли смогу посчитать сколько километров сетевого кабеля используется для всего трафика Azure!

Сетевое подключение — важная часть современной жизни. В Azure сеть — это центр всеобщего взаимодействия. При наличии тысяч физических сетевых устройств и километров сетевых кабелей, соединяющих все в центре обработки данных Azure, вы работаете с *виртуальными* сетевыми ресурсами. Как? В программно-определяемых сетях. Когда вы создаете ВМ или веб-приложение, техник не бегает по ЦОД Azure, чтобы физически подключить кабели и назначить вам IP-адреса (хотя было бы интересно на это посмотреть!). Вместо этого все сетевые ресурсы, определяющие вашу сетевую среду, логически обрабатываются платформой Azure. На рисунке 5.1 показаны компоненты виртуальной сети, которую вы создадите в этой главе.

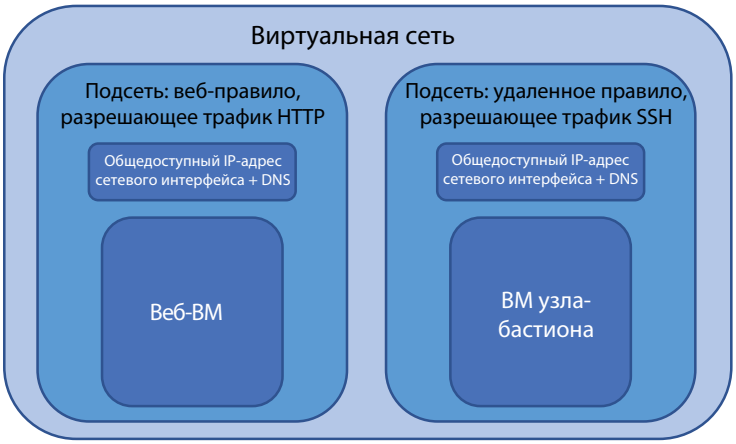


Рис. 5.1. Программно-определяемые сетевые подключения в Azure

Некоторые компоненты сети абстрагируются, если вы используете ресурсы PaaS. Ниже перечислены основные компоненты, используемые для виртуальных машин:

- Виртуальные сети и подсети (включая пулы IP-адресов)
- Виртуальные сетевые адаптеры
- Один или несколько общедоступных IP-адресов
- Внутреннее DNS-имя и необязательные общедоступные DNS-имена для разрешения внешних имен
- Сетевые группы безопасности и правила, защищающие и контролирующие поток сетевого трафика, как это делает обычный брандмауэр

5.1.1 Виртуальные сети и подсети

При создании ВМ в главе 2 вам не нужно было настраивать сетевые параметры. Платформа Azure может создавать эти ресурсы за вас, присваивая им стандартные имена и области IP-адресов. В этом разделе мы создадим сетевые ресурсы и посмотрим, как они объединяются для ВМ.

Попробуйте сейчас

Зачастую сеть легче визуализировать, когда вы видите ее работу. Вы используете портал Azure для начала работы. В конечном итоге вам потребуется довольно много отдельных шагов, но вы ознакомитесь и с возможностями Azure CLI позже в этой главе.

Не слишком беспокойтесь о том, как использовать собственные адресные пространства или DNS-имена прямой сейчас. Чтобы создать виртуальную сеть и подсеть, выполните следующие действия:

1. Зайдите на портал Azure и нажмите кнопку «Создать ресурс» в верхнем левом углу панели.
2. Выберите раздел «Сети» в списке сервисов Marketplace, а затем нажмите кнопку «Виртуальная сеть».
3. Введите имя виртуальной сети, например `vnetmol`.
4. Чтобы было еще интереснее, измените адресное пространство на `10.0.0.0/16`.

Диапазоны IP-адресов

Виртуальные сети охватывают определенный диапазон IP-адресов — адресное пространство. Если вы видели IP-адрес, вероятно, заметили маску подсети: часто что-то вроде `255.255.255.0`. Эта маска подсети часто используется в краткой форме, определяющей ширину диапазона, например `/24`.

По умолчанию в портале Azure используется адресное пространство `/24`. Если вы не слишком разбираетесь в сетях, но хотите увеличить число дополнительных диапазонов IP-адресов, увеличивайте адресное пространство до `/16`. Вы не задаете этот тип IP-адреса напрямую в ВМ — на следующем шаге вы создадите подсеть, охватывающую меньший раздел этого адресного пространства.

Не волнуйтесь, если вы ничего не знаете об адресных пространствах. По большей части вы не будете работать с ними каждый день. Разумная система управления Azure работает так же, как в вашей локальной среде: одна рабочая группа контролирует виртуальные сети Azure, а вы сохраняете свои приложения в заранее созданном пространстве.

5. Создайте группу ресурсов, например `azuremolchapter5`, а затем выберите регион Azure рядом с вами.
6. Укажите имя подсети, например `websubnet`, и введите в поле «Диапазон адресов подсети» значение `10.0.1.0/24`. Этот диапазон входит в широкую виртуальную сеть, указанную ранее. Позже вы добавите еще одну подсеть.
7. Рассмотрим другие варианты, такие как защита от распределенных атак типа «отказ в обслуживании» (DDoS), конечные точки сервиса и брандмауэр Azure. Сейчас оставьте значения по умолчанию, но я надеюсь, этот пример поможет вам понять, какие возможности, помимо базовой виртуальной сети, также доступны.
8. Когда все будет готово, создайте виртуальную сеть и подсеть.

5.1.2 Виртуальные сетевые адаптеры

Теперь, когда вы создали виртуальную сеть и подсеть, необходимо подключить ВМ. Как и на обычном настольном ПК, ноутбуке или планшете, вы подключаетесь к виртуальной сети с помощью виртуального сетевого адаптера (NIC). И никакого бесплатного Wi-Fi! Но в Azure есть размеры ВМ, поддерживающие до 8 адаптеров со скоростью до 32 Гбит/сек. Даже если бы я был хорош в математике, я мог бы сказать вам, что это серьезная пропускная способность!

Наверное, вам интересно, почему мы создаем эти ресурсы заранее. Ведь все это делается при создании ВМ. Факт. Но давайте вернемся на шаг назад и подумаем о долговечности сетевых ресурсов.

Сетевые ресурсы существуют отдельно от ресурсов ВМ и могут проработать дольше самой машины. Эта концепция позволяет создавать фиксированные сетевые ресурсы, а также создавать, удалять и повторно создавать ВМ, поддерживающую те же самые сетевые ресурсы, например IP-адреса и DNS-имена. Представьте себе ВМ для практики или среду разработки и тестирования. Вы можете быстро воспроизвести точно такую же, поскольку изменяется только ВМ.

Попробуйте сейчас

Чтобы создать сетевой адаптер, выполните указанные ниже действия:

1. На портале Azure нажмите кнопку «Создать ресурс» в верхнем левом углу панели управления.
2. Найдите и выберите пункт «Сетевой интерфейс», а затем нажмите «Создать».
3. Введите имя сетевого интерфейса, например `webvnic`. Затем выберите виртуальную сеть и подсеть, которые вы создали в предыдущем упражнении.
4. Я уже упоминал о долговечных ресурсах, а теперь давайте посмотрим, как это работает. Создайте статическое назначение IP-адреса, использующее адрес `10.0.1.4`.

СОВЕТ. ПОЧЕМУ .4? А ГДЕ ЖЕ ПЕРВЫЕ 3 АДРЕСА В АДРЕСНОМ ПРОСТРАНСТВЕ? AZURE РЕЗЕРВИРУЕТ ПЕРВЫЕ 3 IP-АДРЕСА В КАЖДОМ ДИАПАЗОНЕ ДЛЯ СОБСТВЕННОГО УПРАВЛЕНИЯ И МАРШРУТИЗАЦИИ. ПОЭТОМУ .4 — ЭТО ПЕРВЫЙ ИЗ ДОСТУПНЫХ АДРЕСОВ В КАЖДОМ ДИАПАЗОНЕ.

5. «Не создавайте сейчас группу безопасности сети» — мы вернемся к этому через пару минут. Если вы один из знатоков IPv6, установите флажок «Частный IP-адрес (IPv6)» и введите имя. Иначе используйте IPv4.
6. Выберите существующую группу ресурсов из предыдущего упражнения. Затем создайте сетевой адаптер в том же регионе, в котором находится виртуальная сеть.
7. Когда все будет готово, создайте сетевой адаптер.

Разделение ролей в Azure

Вам не нужно создавать другие вычислительные ресурсы в группе ресурсов своей виртуальной сети. Вспомните концепцию системы управления Azure, о которой мы говорили ранее. У вас может быть группа сетевых инженеров, управляющих всеми ресурсами виртуальной сети в Azure. При создании ресурсов для таких приложений, как ВМ, вы создаете и управляете ими в собственных группах ресурсов.

В последующих главах мы рассмотрим ряд функций политик и безопасности в Azure, позволяющих контролировать доступ к определенным ресурсам и возможность их изменения. Суть в том, что если вы не знаете или не хотите знать, к каким ресурсам подключаться, можно подключиться к имеющимся. То же самое относится и к другим инженерам или разработчикам — они видят ресурсы приложения, но не могут редактировать и удалять их.

Эта модель управления в Azure хороша, но избегайте работы в изолированных средах. На крупных предприятиях ваши действия могут ограничиваться другими отделами. Но такие поставщики облачных вычислений, как Azure, обладают огромным преимуществом: они ускоряют развертывание приложений, поскольку не нужно ждать подготовки и настройки физических сетевых ресурсов. Запланировав создание и настройку сетевых ресурсов Azure, вы сможете легко создавать и контролировать ресурсы в своих приложениях.

5.1.3 *Публичный IP-адрес и DNS-разрешение*

Сейчас никто не может получить доступ к вашим ресурсам, поскольку для них не заданы публичные IP-адреса или DNS-имена. Повторюсь, давайте следовать принципу долгосрочных ресурсов при создании IP-адреса и DNS-имени, а затем назначать их своему сетевому интерфейсу.

Попробуйте сейчас

Чтобы создать публичный IP-адрес и запись DNS для своего сетевого интерфейса, выполните приведенные ниже действия:

1. На портале Azure нажмите кнопку «Создать ресурс» в верхнем левом углу панели управления.
2. Найдите и выберите публичный IP-адрес, а затем нажмите кнопку «Создать».
3. Создайте базовый номер SKU и IPv4-адрес. SKU «Стандартный» и IPv6-адреса предназначены для использования в подсистеме балансировки нагрузки, которую мы рассмотрим в главе 8. Пока что не волнуйтесь об отличиях!
4. Введите имя, например webpublicip, которое использует динамическое назначение.

Типы назначения IP-адресов

При динамическом назначении публичный IP-адрес выделяется во время запуска ВМ. При остановке ВМ его выделение отменяется. Здесь есть несколько важных нюансов:

- У вас не будет публичного IP-адреса, пока вы не назначите его для ВМ и не запустите ее.
- Публичный IP-адрес может измениться, если остановить, освободить или запустить ВМ.

Статическое назначение указывает, что у вас есть публичный IP-адрес, выделенный без связанной ВМ, и этот адрес не изменится. Это удобно для сценариев, в которых используется SSL-сертификат, сопоставленный с IP-адресом, или пользовательское DNS-имя и запись, указывающие на этот IP-адрес.

Сейчас вы используете одну ВМ. В производственной среде рекомендуется запускать приложение на нескольких ВМ с предстоящей им подсистемой балансировки нагрузки. В подобном случае публичный IP-адрес назначается подсистеме балансировки нагрузки и обычно создает статическое назначение в этой точке.

5 Введите уникальное DNS-имя. Это имя формирует полное доменное имя (FQDN) вашего ресурса на основе региона его создания в Azure. Например, если создать DNS-имя azuremol в восточной части США, полное доменное имя будет выглядеть так: azuremol.eastus.cloudapp.azure.com.

Записи DNS

А что насчет пользовательского DNS-имени? Стандартное FQDN не совсем понятное! Используйте статический публичный IP-адрес, а затем создайте запись CNAME в зарегистрированной зоне DNS. Вы сохраняете контроль над записью DNS и можете создавать неограниченное число записей для своих приложений.

Например, в зоне DNS manning.com вы могли бы создать запись CNAME для azuremol, указывающую на статический публичный IP-адрес в Azure. В этом случае пользователь получает доступ к приложению через сайт azuremol.manning.com. Этот адрес намного удобнее, чем webmol.eastus.cloudapp.azure.com!

6 Выберите существующую группу ресурсов из предыдущего упражнения. Затем создайте публичный IP-адрес в том же регионе, в котором находится виртуальная сеть.

7 Когда все будет готово, создайте публичный IP-адрес.

8 Свяжите публичный IP-адрес и метку DNS-имени с сетевым интерфейсом, созданным в разделе 5.1.2. Найдите и нажмите кнопку «Группа ресурсов» на левой панели навигации портала Azure. Затем выберите группу ресурсов, в которой находятся созданные вами сетевые ресурсы, например azuremolchapter5.

9 Выберите публичный IP-адрес из списка ресурсов, а затем нажмите кнопку «Связать».

10 Выберите связь с сетевым интерфейсом (но обратите внимание на то, с чем еще вы можете связать публичный IP-адрес). Затем выберите созданный вами сетевой интерфейс, например webvnic.

Через несколько секунд окно публичного IP-адреса обновится и покажет, что IP-адрес теперь связан с вашим сетевым интерфейсом. Если вы выбрали тип назначения «Динамический», IP-адрес остается пустым, как показано на рисунке 5.2. Помните, что публичный IP-адрес выделяется после того, как запускается связанная ВМ.

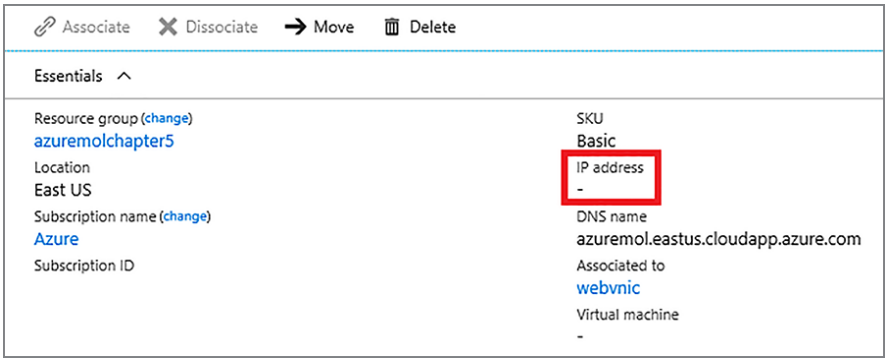


Рис. 5.2. Публичный IP-адрес связан с сетевым интерфейсом. При динамическом назначении публичный IP-адрес не отображается, пока ВМ не создана и не включена.

5.2 Защита и контроль трафика с помощью групп безопасности сети

Время контрольной проверки: следует ли подключать ВМ к Интернету без брандмауэра, контролирующего и ограничивающего поток трафика? Если вы ответили «Конечно, почему бы и нет?», вероятно, в оставшееся обеденное время вам следует немного почитать о безопасности сети на просторах всемирной паутины!

Я надеюсь, вы четко ответили «*Нет!*». К сожалению, слишком высока вероятность автоматической кибератаки на вашу ВМ после ее включения. Всегда следуйте рекомендациям по обновлению программного обеспечения для ОС и приложения, а также открывайте ВМ для сетевого трафика только при необходимости. На обычном компьютере под управлением macOS или Windows установлен встроенный брандмауэр, и все подобающие локальные сети на моей памяти были оснащены сетевым брандмауэром между Интернетом и внутренней сетью. В Azure правила брандмауэра и трафика обеспечиваются группами безопасности сети.

5.2.1 Создание группы безопасности сети

В Azure группа безопасности сети (NSG) логически применяет набор правил к сетевым ресурсам. Эти правила фильтруют входящий и исходящий трафик ВМ. Вы определяете разрешенные порты, протоколы и IP-адреса и направление доступа к ним. Эти группы правил применимы к одному сетевому интерфейсу или ко всей подсети. Такая гибкость позволяет точно контролировать, как и когда правила защитят ваше приложение.

На рисунке 5.3 показана логическая схема движения входящего сетевого пакета при его прохождении через NSG. Такой же процесс применяется для исходящих пакетов. Узел Azure не различает интернет-трафик и трафик из других источников в среде Azure, например из другой подсети или виртуальной сети. К любому входящему сетевому пакету применяются правила NSG для входящего трафика, а к любому исходящему — правила NSG для исходящего трафика.

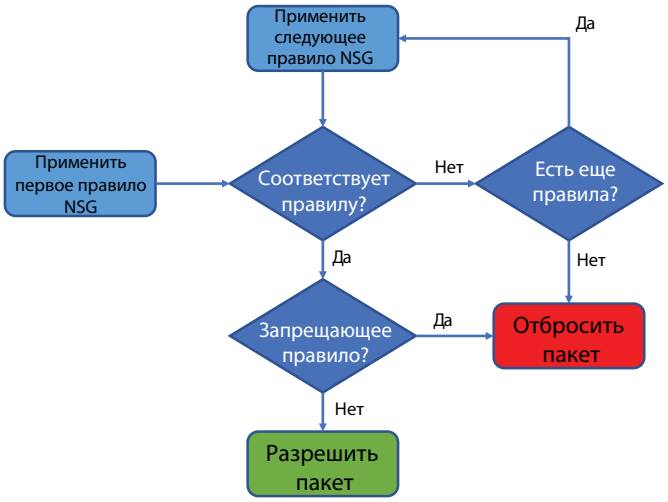


Рис. 5.3. Входящие пакеты проверяются, и каждое правило NSG применяется в порядке приоритета. Если согласно правилу выполняется действие «Разрешить» или «Запретить», пакет перенаправляется на ВМ либо удаляется.

Вот что происходит с каждым сетевым пакетом:

1. Применяется первое правило NSG.
2. Если правило не соответствует пакету, загружается следующее правило и так далее, пока не останется правил. Затем применяется стандартное правило для удаления пакета.
3. При соответствии правилу проверьте, запрещается ли пакет действием. Если да, то пакет удаляется.
4. И наоборот, если правило разрешает пакет, он передается в ВМ.

Затем вы создадите группу безопасности сети, чтобы эти концепции воплотились в реальность.

Попробуйте сейчас

Чтобы создать группу безопасности сети, выполните следующие действия:

1. На портале Azure нажмите кнопку «Создать ресурс» в верхнем левом углу панели управления.
2. Найдите и выберите раздел «Группа безопасности сети», а затем нажмите кнопку «Создать».
3. Введите имя, например `webnsg`, и выберите существующую группу ресурсов.

Вот и все! Основная настройка NSG выполняется при создании правил фильтрации. В разделе 5.2.2. мы обсудим, как это сделать, и применим NSG на практике.

5.2.2 *Связывание группы безопасности сети с подсетью*

NSG не защитит вашу ВМ при отсутствии правил. Кроме того, необходимо связать ее с подсетью так же, как вы связывали публичный IP-адрес с сетевым интерфейсом. Для начала свяжем вашу NSG с подсетью.

Попробуйте сейчас

Чтобы связать подсеть виртуальной сети с группой безопасности сети, выполните приведенные ниже действия:

1. Найдите и нажмите кнопку «Группа ресурсов» на левой панели навигации портала Azure. Затем выберите группу ресурсов, которая содержит созданные вами сетевые ресурсы, например azuremolchapter5.
2. Выберите NSG, например webnsg.
3. Слева в разделе «Параметры» есть пункты «Сетевые интерфейсы» и «Подсети». Выберите «Подсети».
4. Нажмите кнопку «Связать», выберите виртуальную сеть и подсеть, созданные ранее. Нажмите кнопку «ОК», чтобы связать группу безопасности сети с подсетью.

   Гибкость NSG означает, что вы можете связать одну NSG с несколькими подсетями в различных виртуальных сетях. Это сопоставление «один ко многим», позволяющее определить основные правила безопасности сети, которые применяются к широкому диапазону ресурсов и приложений.

   Теперь вы можете узнать, как выглядит ваша NSG и какие правила применяются по умолчанию.
5. Слева в своей NSG выберите пункт «Правила безопасности для входящего трафика». Здесь правила безопасности не указаны; как минимум нет ни одного из созданных вами.
6. Выберите раздел «Правила по умолчанию», чтобы увидеть, что создала для вас платформа Azure. См. рисунок 5.4.

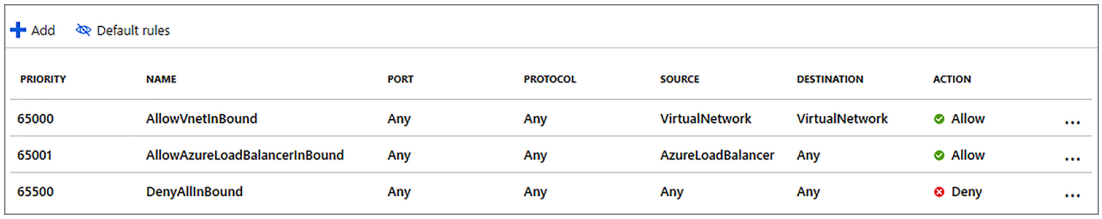


Рис. 5.4. По умолчанию создаются правила безопасности, разрешающие трафик во внутренней виртуальной сети и трафик подсистемы балансировки нагрузки, но запрещающие весь остальной трафик.

Для вас созданы три стандартных правила. Важно разбираться в них:

- *AllowVnetInBound* — разрешает весь внутренний трафик в виртуальной сети. Если в вашей виртуальной сети есть несколько подсетей, по умолчанию трафик разрешен и не фильтруется.
- *AllowAzureLoadBalancerInBound* — разрешает весь трафик из подсистемы балансировки нагрузки Azure в ВМ. Если вы размещаете такую подсистему между ВМ и Интернетом, это правило гарантирует, что трафик из подсистемы балансировки нагрузки достигнет ваших ВМ, например для отслеживания пульса.
- *DenyAllInBound* — последнее применимое правило. Удаляет абсолютно все входящие пакеты. Если нет предыдущих правил «Разрешить», это правило по умолчанию удаляет весь трафик. Т. е. нужно просто разрешить необходимый вам трафик. Весь остальной трафик блокируется.

Важен приоритет правила NSG. Если применяется правило «Разрешить» или «Запретить», никакие дополнительные правила уже не используются. Правила применяются в порядке возрастания значений приоритета. Например, правило с приоритетом 100 применяется перед правилом с приоритетом 200.

Как и в системе управления ресурсами Azure, которую мы обсуждали ранее, указанные правила NSG могут быть заранее созданы и применены к этой подсети. Вы создаете ВМ и запускаете приложения, а NSG делает кто-то другой.

Важно понимать, как проходит трафик в случае возникновения проблем. Несколько инструментов в Azure помогут определить, почему трафик не попадает в ваше приложение, хотя вы думаете, что все в порядке!

5.2.3 Создание правил фильтрации для группы безопасности сети

Теперь, когда NSG связана с подсетью и мы рассмотрели стандартные правила, давайте создадим базовое правило NSG, разрешающее HTTP-трафик.

Попробуйте сейчас

Чтобы создать свои правила в группе безопасности сети, выполните приведенные ниже действия:

1. Чтобы создать правило NSG в предыдущем окне портала Azure, нажмите кнопку «Добавить» в разделе «Правила безопасности для входящего трафика».
2. У вас есть 2 варианта создания правил: базовый и расширенный. Чтобы быстро создать готовые правила, выберите «Базовый» в верхней части окна.
3. Выберите HTTP в раскрывающемся меню «Сервис». Здесь вы найдете много стандартных сервисов, например SSH, RDP и MySQL. При выборе сервиса применяется соответствующий диапазон портов. В этом случае это порт 80. По умолчанию для основных правил трафик разрешен.
4. Каждому правилу присваивается «Приоритет». Чем меньше число, тем выше приоритет. Оставьте стандартный низкий приоритет, например 100.
5. Примите имя по умолчанию или введите новое. Затем нажмите кнопку «ОК».

5.3 Создание примера веб-приложения с защищенным трафиком

Итак, мы создали виртуальную сеть и подсеть. Затем создали сетевой интерфейс и связали публичный IP-адрес и метку DNS-имени. Создали NSG и применили ее ко всей подсети, а также создали правило NSG, чтобы разрешить HTTP-трафик. Нам не хватает только одного: ВМ.

5.3.1 Создание сетевых подключений удаленного доступа

В производственной среде вам не следует открывать удаленный доступ, например через SSH или RDP, к ВМ, выполняющей приложения. Обычно у вас есть отдельная ВМ Jumpbox, и вы подключаетесь к ней через Интернет, а затем получаете доступ к дополнительным ВМ через внутреннее соединение. До сих пор вы создавали все ресурсы виртуальной сети на портале Azure. Перейдем к Azure CLI, чтобы увидеть, как можно быстро создать эти ресурсы из командной строки.

Попробуйте сейчас

Первую NSG вы создали на портале Azure. Чтобы создать другую NSG с помощью Azure CLI, выполните приведенные ниже действия:

1. Щелкните значок Cloud Shell в верхней части панели на портале Azure. Убедитесь, что открывается Bash, а не PowerShell.
2. Создайте дополнительную NSG в существующей группе ресурсов. Как и в предыдущих главах, обратная косая черта (\) в приведенных ниже примерах команд используется для разрыва строк, и ее можно не вводить. Укажите имя, например remotensg:

```
az network nsg create \
  --resource-group azuremolchapter5 \
  --name remotensg
```

3. Создайте правило в новой NSG, *разрешающее* порт *22*. Укажите группу ресурсов и NSG, созданную вами на предыдущем шаге, а также имя, например allowssh:

```
az network nsg rule create \
  --resource-group azuremolchapter5 \
  --nsg-name remotensg \
  --name allowssh \
  --protocol tcp \
  --priority 100 \
  --destination-port-range 22 \
  --access allow
```

4. Создайте подсеть для удаленной ВМ. Укажите имя подсети, например remotesubnet, а также префикс адреса в диапазоне виртуальной сети, например `10.0.2.0/24`. Подключите NSG, созданную на шаге 3, к такой подсети, как remotensg:

```
az network vnet subnet create \
  --resource-group azuremolchapter5 \
```

```
--vnet-name vnetmol \
--name remotesubnet \
--address-prefix 10.0.2.0/24 \
--network-security-group remotensg
```

3 команды — это все, что требуется, чтобы создать подсеть, NSG и правило. Вы уже начали понимать силу Azure CLI? Программа Azure PowerShell не менее мощная, поэтому вам не обязательно создавать все ресурсы только на портале Azure. Далее в книге мы будем чаще использовать Azure CLI, а не портал.

5.3.2 Создание ВМ

Все сетевые компоненты установлены. Теперь мы можем создать 2 ВМ. Одну ВМ создадим в подсети, разрешающей HTTP-трафик, чтобы вы могли установить веб-сервер. Вторую — в подсети, разрешающей SSH. Таким образом, мы получим Jumpbox для защиты среды приложения и репликации производственного развертывания. Рисунок 5.5 напомнит вам о том, что вы создаете.

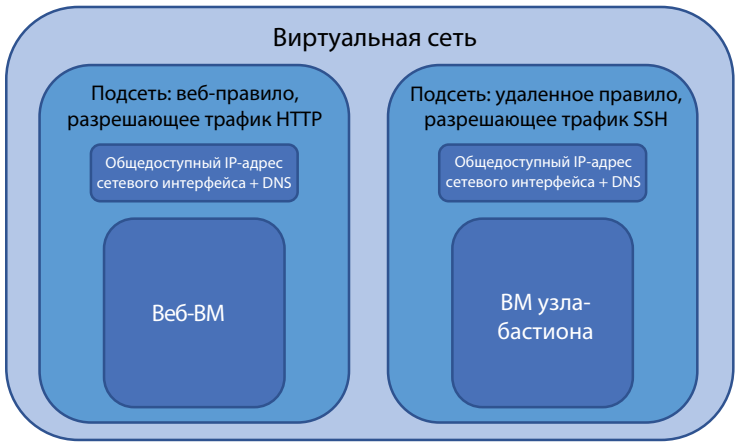


Рис. 5.5. Вы объединяете две подсети, NSG, правила, сетевые интерфейсы и ВМ. Это очень похоже на производственное развертывание, когда одна ВМ запускает веб-сервер и открыта для публичного трафика, а другая ВМ в отдельной подсети используется для удаленных подключений к остальной среде приложения.

При создании ВМ можете использовать интерфейс виртуальной сети, созданный вами ранее. Если не указан этот сетевой ресурс, Azure CLI создаст для вас виртуальную сеть, подсеть и сетевой адаптер, используя встроенные стандартные значения. Это удобно, если нужно быстро создать ВМ. Но мы следуем принципу долгосрочных сетевых ресурсов, открытых для других разработчиков и позволяющих создать ВМ.

Попробуйте сейчас

Чтобы создать виртуальные машины Jumpbox в Azure CLI, выполните приведенные ниже действия:

1. Создайте первую ВМ для своего веб-сервера и укажите имя, например webvm. Подключите сетевой интерфейс, например webvnic, и выберите такой образ, как UbuntuLTS. Укажите имя пользователя, например azuremol. Заключительная команда `--generate-ssh-keys` добавляет к ВМ ключи SSH, созданные вами в главе 2:

```
az vm create \
  --resource-group azuremolchapter5 \
  --name webvm \
  --nics webvnic \
  --image UbuntuLTS \
  --size Standard_B1ms \
  --admin-username azuremol \
  --generate-ssh-keys
```

2. Создайте вторую ВМ для Jumpbox. В этом примере показано, как использовать существующую подсеть и NSG и как разрешить Azure CLI создание сетевого интерфейса и выполнение соответствующих подключений. Этой командой вы также создаете публичный IP-адрес, например remotepublicip:

```
az vm create \
  --resource-group azuremolchapter5 \
  --name remotevm \
  --vnet-name vnetmol \
  --subnet remotesubnet \
  --nsg remotensg \
  --public-ip-address remotepublicip \
  --image UbuntuLTS \
  --size Standard_B1ms \
  --admin-username azuremol \
  --generate-ssh-keys
```

Обе команды выводят публичный IP-адрес. Запишите эти IP-адреса. Если в следующем упражнении вы попытаетесь подключиться к первой ВМ для веб-сервера по протоколу SSH, произойдет сбой. Почему? Вы можете подключиться по протоколу SSH к удаленной ВМ, так как вы создали правило NSG, разрешающее только HTTP-трафик для ВМ веб-сервера.

5.3.3 Использование агента SSH для подключения к ВМ

Позвольте представить вам один волшебный инструмент, позволяющий правильно использовать Jumpbox и подключаться к веб-ВМ через виртуальную сеть Azure: он называется *агент SSH*. Он относится только к ВМ Linux, поэтому, если вы работаете в основном с ВМ Windows и удаленными рабочими столами (RDP), не переживайте, что раньше ничего не слышали об этом. Вы можете создавать RDP-подключения к ВМ Windows из своего Jumpbox, используя локальные учетные данные для удаленного подключения или учетные данные домена, если сервер настроен соответствующим образом.

Агент SSH может хранить ваши ключи SSH и при необходимости пересылать их. Еще в главе 2, когда мы создавали пару открытых ключей SSH, я говорил об открытом и закрытом ключах. Закрытый ключ остается на вашем компьютере. Только открытый ключ копируется на удаленные ВМ. Хотя открытый ключ добавлен к двум

созданным ВМ, вы не сможете выполнять передачу по SSH в Jumpbox, а затем на ВМ в Интернете. Почему? Потому что в Jumpbox нет копии вашего закрытого ключа. SSH-подключение к Jumpbox невозможно, поскольку нет закрытого ключа — пары для открытого ключа на ВМ в Интернете — для аутентификации.

Закрытый ключ следует оберегать, поэтому не ищите простой путь, копируя его в Jumpbox. Есть вероятность, что пользователь с доступом к Jumpbox завладеет копией вашего закрытого ключа, а затем будет изображать вас везде, где нужен этот ключ. Здесь-то и пригодится агент SSH.

Если запустить агента SSH в сеансе Cloud Shell, вы сможете добавить в него ключи SSH. Чтобы создать SSH-подключение к Jumpbox, укажите, что этот агент используется для туннелирования сеанса. Это позволит эффективно проверять на Jumpbox закрытый ключ, даже не копируя его. Когда выполняется подключение SSH из Jumpbox к ВМ в Интернете, агент SSH туннелирует ваш закрытый ключ через Jumpbox и позволяет успешно пройти аутентификацию.

Попробуйте сейчас

Чтобы использовать SSH с ВМ Jumpbox, выполните приведенные ниже действия:

1. В Cloud Shell запустите агент SSH:

   ```
   eval $(ssh-agent)
   ```

2. Добавьте ключ SSH, созданный в главе 2, в агент:

   ```
   ssh-add
   ```

3. Подключитесь к ВМ Jumpbox по протоколу SSH. Укажите использование агента SSH с параметром `-A`. Введите публичный IP-адрес, который был показан при создании ВМ Jumpbox:

   ```
   ssh -A azuremol@<publicIpAddress>
   ```

4. Вы впервые создали SSH-подключение к ВМ Jumpbox, поэтому примите запрос на подключение с ключами SSH.
5. Помните, как вы назначали статический частный IP-адрес для ВМ в Интернете в разделе 5.1.2? Этот статический адрес значительно упрощает SSH-подключение к нему. SSH-подключение к ВМ в Интернете:

   ```
   ssh 10.0.1.4
   ```

6. Примите запрос на продолжение SSH-подключения. SSH-агент туннелирует ваш закрытый ключ SSH через Jumpbox и разрешает успешно подключиться к веб-ВМ. Что теперь? А теперь выполним практическое задание, чтобы посмотреть, как это работает!

5.4 Практическое задание: установка и тестирование веб-сервера LAMP

Вы хорошо потрудились в этой главе. В небольшом практическом задании вы закрепите установку веб-сервера и увидите, как правило NSG работает на вашей ВМ:

1. *Установите простой веб-сервер Linux*. Вспомните главу 2, когда вы создавали SSH-подключение к ВМ, а затем устанавливали пакет веб-сервера LAMP с помощью `apt`. Через SSH-подключение к вашей веб-ВМ, созданной в разделе 5.3.2, установите и настройте стандартный веб-стек LAMP.
2. *Перейдите на стандартный веб-сайт*. После установки веб-стека LAMP откройте браузер по метке DNS-имени, введенной вами при создании публичного IP-адреса в разделе 5.1.3. В моем примере это azuremol.eastus.cloudapp.azure.com. Вы также можете использовать публичный IP-адрес, выведенный при создании веб-ВМ. Запомните! Публичный IP-адрес отличается от адреса ВМ Jumpbox, к которой вы подключаетесь через SSH.

Часть 2

Высокая доступность и масштабирование

Теперь немного повеселимся. Теперь, когда вы разобрались с основными ресурсами в Azure, можно перейти к избыточности, балансировке нагрузки и географически распределяемым приложениям. Именно здесь находится все самое интересное. Я надеюсь, что в предстоящих темах вы найдете решения и рекомендации для использования в реальных развертываниях. Azure предлагает потрясающие возможности для глобальной репликации данных, распределения трафика клиентов к ближайшему экземпляру приложения и автоматического масштабирования по запросу. В этих возможностях кроется сила облачных вычислений и истинная ценность для вашей работы.

6 Azure Resource Manager

Наверняка вы хотите как можно быстрее наладить среду приложений и перейти к фактическому развертыванию. Во многих ИТ-средах начинают формироваться команды управления процессом разработки, которые тесно сотрудничают в соответствии с термином *DevOps*, — модным выражением многочисленных блогов и конференций.

В культуре DevOps нет ничего принципиально нового или революционного, однако часто разные команды работают вместе не так, как следовало бы. Современные средства дали импульс модели DevOps благодаря решениям непрерывной интеграции и развертывания (CI/CD), которые полностью автоматизируют развертывание сред приложений с помощью единого кода от разработчика. Как правило, конвейеры CI/CD создаются и поддерживаются командой эксплуатации, что значительно ускоряет тестирование и развертывание обновлений приложений для разработчиков.

Модель развертывания Azure Resource Manager незаменима для создания и выполнения ресурсов, даже если вы еще не осознали это. Resource Manager — это модель создания и развертывания ресурсов, а также процессов автоматизации и шаблонов, которые управляют развертываниями. В этой главе рассматривается использование функций Resource Manager, например элементов управления доступом и блокировок, согласованных развертываний шаблонов и автоматизированных многоуровневых развертываний.

6.1 Модель Azure Resource Manager

При создании виртуальной машины или веб-приложения в предыдущих главах сначала создавалась группа ресурсов как основная конструкция для включения в нее всех ресурсов. Группа ресурсов — один из главных аспектов для всех ресурсов: виртуальная машина, веб-приложение, виртуальная сеть или таблица хранилища не могут находиться вне группы. Но группа ресурсов — это намного больше, чем просто их расположение. В этом разделе мы рассмотрим базовую модель Azure Resource Manager и ее значение для создания и выполнения приложений.

6.1.1 Проектирование на основе жизненного цикла приложений

Надеюсь, вы не станете создавать приложение, о котором тут же забудете. Обычно нужно разрабатывать и развернуть обновления, устанавливать новые пакеты, добавлять новые ВМ и создавать дополнительные слоты развертывания веб-приложений. Возможно, потребуется изменять параметры виртуальной сети и IP-адреса. В предыдущих главах я упоминал, что вашими виртуальными сетями в Azure может управлять другая команда. Пора начинать думать о работе в глобальном масштабе с точки зрения жизненного цикла приложений и управления.

Есть несколько основных подходов к группированию ресурсов в Azure:

- *Все ресурсы для данного приложения находятся в одной и той же группе*. Как показано на рис. 6.1, этот подход хорошо работает для небольших приложений или сред разработки и тестирования. Если не нужно совместно использовать большие сетевые пространства и можно управлять хранилищем индивидуально, целесообразно создать все ресурсы в одном расположении и управлять обновлениями и изменениями конфигурации посредством одной операции.

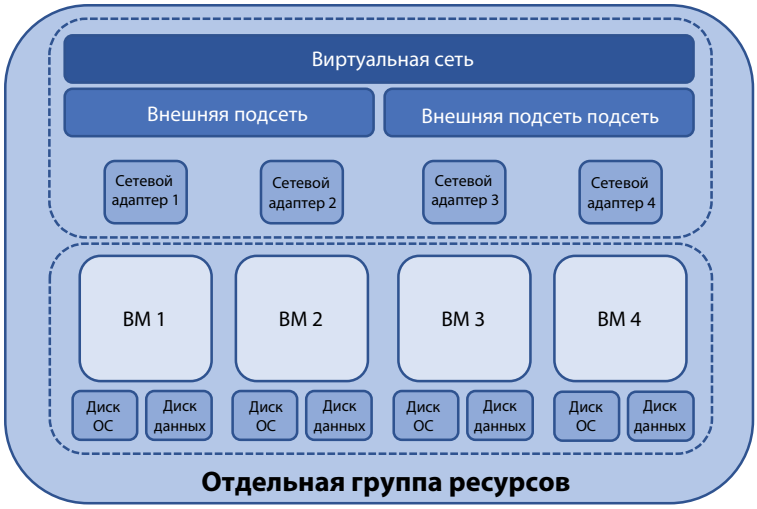


Рис. 6.1. Один из способов создания приложений в Azure состоит в том, что все ресурсы, связанные с развертыванием приложения, создаются в одной группе ресурсов и управляются как одна сущность.

- *Сходные ресурсы группируются по функции в одной и той же группе ресурсов*. Как показано на рисунке 6.2, этот подход чаще встречается в больших приложениях и средах. Приложение может находиться в группе ресурсов только с виртуальными машинами и вспомогательными компонентами приложения. Ресурсы виртуальной сети и IP-адреса могут располагаться в иной группе ресурсов, защищенной и управляемой другой командой инженеров.

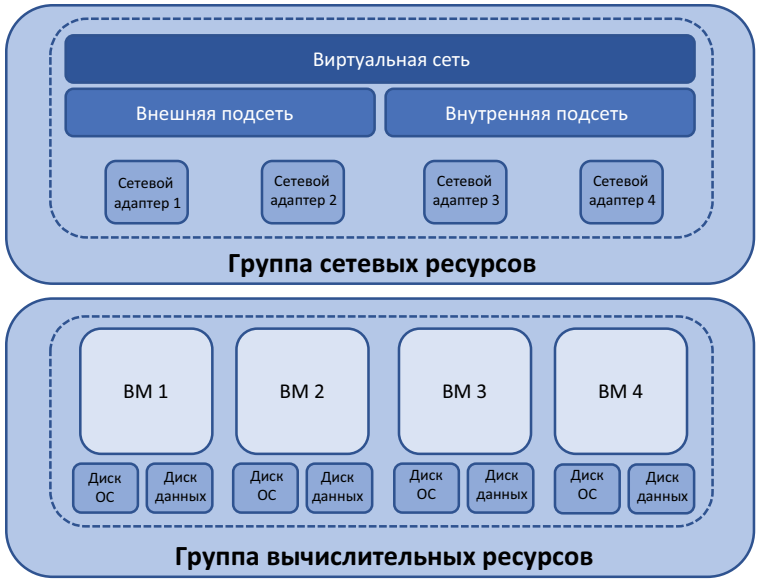


Рис. 6.2. Альтернативный подход заключается в создании и группировании ресурсов на основе их роли. Популярный пример: все основные сетевые ресурсы находятся в одной группе ресурсов, а основные вычислительные ресурсы приложения — в другой. Виртуальные машины в группе вычислительных ресурсов могут получать доступ к сетевым ресурсам в отдельной группе, но эти два набора ресурсов можно контролировать и защищать независимо друг от друга.

Зачем нужны разные подходы? Причина не только в обеспечении занятости и самодостаточных подразделениях, в которых так нравится работать некоторым командам. Преимущественно речь идет о том, как нужно управлять базовыми ресурсами. В небольших средах и приложениях, где все ресурсы находятся в одной и той же группе, вы отвечаете за все процессы в среде. Этот подход также хорошо работает для сред разработки и тестирования, где все упаковано вместе. Любые изменения виртуальной сети влияют только на ваше приложение и группу ресурсов.

В действительности сети меняются редко. Зачастую диапазоны адресов хорошо определены и спланированы для сосуществования в разных расположениях Azure и в офисах по всему миру. Логично поместить сетевые компоненты в их собственную группу ресурсов. Сеть управляется отдельно от приложения. Таким же образом может управляться и обновляться хранилище. В подобном разделении ресурсов нет ничего плохого, если только ИТ-персонал не погрязнет в разрозненности, что приведет к утрате сотрудничества.

Разделять ресурсы приложений выгодно еще и потому, что это не накладывает ограничений при изменениях и обновлениях. Раз в группе ресурсов нет сетевых компонентов, вам не нужно беспокоиться о них во время обновления приложений.

6.1.2 *Защита и управление ресурсами*

К каждому ресурсу можно применить разные разрешения безопасности. Эти политики определяют, кто и что может делать. Подумайте, разрешать ли стажеру перезапускать веб-приложение или удалять диски данных виртуальной машины? Стоит ли создавать новую подсеть виртуальной сети для своих приятелей в сетевой группе? Скорее всего, нет. В Azure есть 4 основные роли, которые можно назначать ресурсам (подобно разрешениям для файлов):

- *Владелец* — полный контроль, по сути администратор
- *Участник* — может полностью управлять ресурсом, кроме изменений защиты и назначения ролей.
- *Читатель* — может просматривать всю информацию о ресурсе, но не может вносить никаких изменений.
- *Администратор доступа пользователей* — может назначать или удалять доступ к ресурсам.

Управление доступом на основе ролей (RBAC) — основная функция ресурсов Azure, которая автоматически интегрируется с учетными записями пользователей в подписках. Подумайте о разрешениях для файлов на своем обычном компьютере. Основные: чтение, запись и выполнение. Комбинируя их, можно создавать различные наборы разрешений для каждого пользователя или группы на компьютере. При работе с сетевыми общими папками, разрешения — это распространенный инструмент управления доступом. RBAC управляет доступом к ресурсам в Azure с помощью тех же строк, что назначают разрешения для сетевых папок или файлов на локальном компьютере (см. рисунок 6.3).

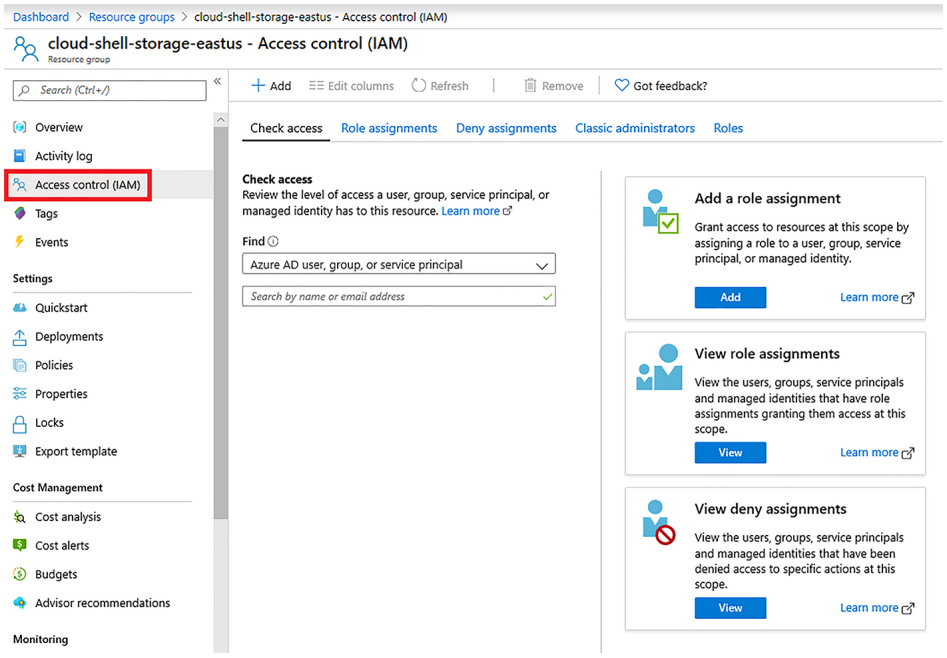


Рис. 6.3. В управлении доступом для каждого ресурса Azure перечисляются текущие назначения. Вы можете добавить назначения или нажать кнопку «Роли», чтобы просмотреть сведения о доступных наборах разрешений.

Попробуйте сейчас

Откройте портал Azure в веб-браузере и выберите любой ресурс, например cloud-shell-storage. Нажмите кнопку «Управление доступом (IAM)» (см. рис. 6.3). Просмотрите текущие назначения ролей. Узнайте, как добавить назначение роли, и изучите все доступные назначения. Значок «Информация» для каждой роли показывает, какие разрешения назначены.

Некоторые из доступных ролей применимы к определенным ресурсам, в том числе следующим:

- Участник виртуальной машины
- Участник веб-сайта
- Участник сети

Догадываетесь, что они означают? Это применение основной роли платформы «Участник» к определенному типу ресурсов. Данный пример использования основан на концепции управления сходными ресурсами. Предположим, вам назначена роль участника виртуальной машины или участника веб-сайта. Затем вы будете контролировать все виртуальные машины или веб-приложения, созданные в этой группе ресурсов. Но вы не сможете управлять сетевыми ресурсами, которые полностью находятся в другой группе ресурсов.

6.1.3 Защита ресурсов с помощью блокировок

Подход RBAC на основе разрешений — отличный способ ограничить доступ выбранных пользователей к определенным ресурсам. Но ошибки все равно возможны. Есть причина, по которой вы обычно не входите на сервер как администратор или привилегированный пользователь. Одно неправильное нажатие клавиши или щелчок мышью могут ошибочно удалить ресурсы. Даже если у вас есть резервные копии (у вас же есть резервные копии, и вы регулярно тестируете их, не так ли?), восстановление — это трудоемкий процесс, который может снизить продуктивность или доходы для бизнеса. В главе 13 вы узнаете подробнее о способах защиты данных с помощью сервисов архивации, восстановления и репликации Azure.

Еще одна встроенная функция модели Resource Manager — блокировка ресурсов. К каждому ресурсу можно применить блокировку, разрешающую доступ только для чтения или запрещающую удаление. Блокировка удаления особенно полезна, поскольку можно запросто удалить не ту группу ресурсов. После того как вы запустите операцию удаления и платформа Azure примет ваш запрос, вернуться назад или отменить операцию будет невозможно.

Для рабочих нагрузок в рабочей среде я предлагаю заблокировать удаление основных ресурсов. Эти блокировки действуют только на уровне ресурсов и платформы Azure, а не данных в составе ресурсов. Например, вы можете удалять файлы в виртуальной машине или таблицы в базе данных. Блокировка ресурсов Azure будет применяться только при попытке удалить всю виртуальную машину или базу данных SQL Azure. Когда блокировка впервые сработает и не позволит ошибочно удалить группу ресурсов, вы скажете мне спасибо.

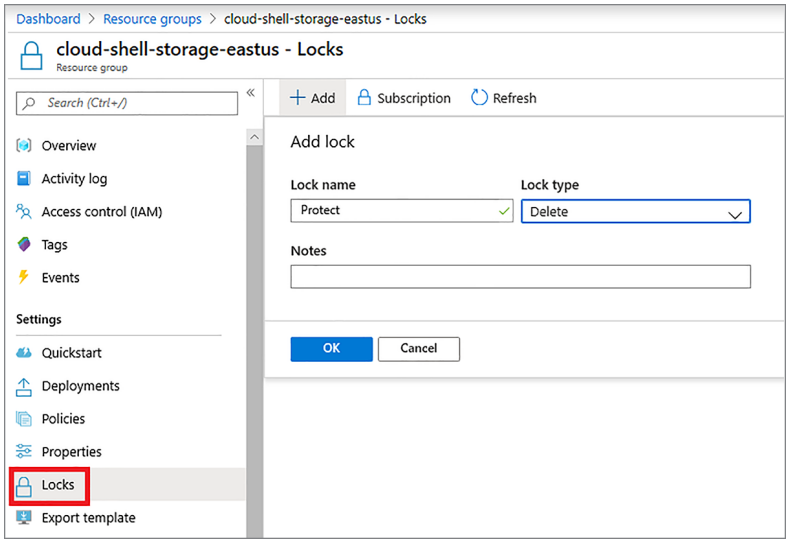


Рис. 6.4. Создание блокировки ресурсов на портале Azure.

Попробуйте сейчас

Чтобы увидеть блокировку ресурсов Azure в действии (см. рис. 6.4), выполните приведенные ниже действия:

1. Откройте портал Azure в веб-браузере и выберите любую группу ресурсов, например cloud-shell-storage.
2. Выберите «Блокировки» в левой части портала.
3. Введите имя блокировки, например «Защита», и выберите «Удалить» в раскрывающемся меню «Тип блокировки». Нажмите кнопку «ОК». Новая блокировка появится в списке.
4. Нажмите «Обзор» для группы ресурсов и попытайтесь удалить ее. Введите имя группы ресурсов, чтобы подтвердить ее удаление (это хорошая подсказка, гарантирующая, что вы удаляете правильный ресурс).
5. Нажмите кнопку «Удалить» и просмотрите появившееся сообщение об ошибке, чтобы узнать, как блокировка не позволила Azure удалить ресурс.

6.1.4 *Контроль и группирование ресурсов с помощью тегов*

Последняя функция в модели Azure Resource Manager, о которой я хочу рассказать, — это *теги*. Нет ничего нового или особенного в маркировке ресурсов в Azure, но этот принцип управления часто упускается из виду. С помощью тегов в Azure можно описать свойства ресурсов, например приложение и среда (разработки или производственная), к которым относится ресурс, и отдел, за него отвечающий.

Затем на основе тегов можно назначить ресурсам блокировки либо роли RBAC или создать отчет о стоимости и потреблении ресурсов. Теги не уникальны для группы ресурсов и могут повторно использоваться в рамках подписки. К одному ресурсу можно применить до 50 тегов, что повышает гибкость маркировки и последующей фильтрации маркированных ресурсов.

Попробуйте сейчас

Чтобы увидеть теги ресурсов Azure в действии, выполните приведенные ниже действия:

1. Откройте портал Azure в веб-браузере и выберите любой ресурс, например cloud-shell-storage. Вы можете пометить саму группу ресурсов, но в этом упражнении не выбирайте ее.
2. Выбрав ресурс, нажмите кнопку «Теги», как показано на рисунке 6.5.
3. Введите имя, например `workload`, и значение, например `development`.
4. Нажмите кнопку «Сохранить».
5. Откройте Cloud Shell.
6. Чтобы отфильтровать ресурсы по тегам, используйте команду `az resource list` с параметром `--tag`. Введите ваше собственное имя и значение следующим образом:

```
az resource list --tag workload=development
```

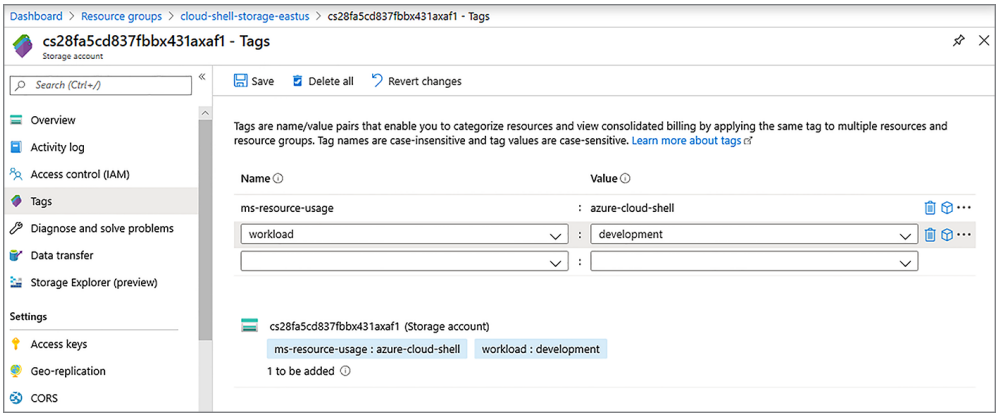


Рис. 6.5. Вы можете создать до 50 тегов «имя:значение» для каждого ресурса или группы ресурсов Azure.

6.2 *Шаблоны Azure Resource Manager*

До сих пор вы создавали небольшое число ресурсов Azure за один раз. Для этого вы использовали портал Azure или интерфейс Azure CLI. Хотя мы не рассматривали модуль Azure PowerShell, я говорил о нем в первой главе, и он доступен в оболочке Cloud Shell. Возможно, вы уже поработали с ним без меня. Это нормально, я не чувствую себя «за бортом». Как упоминалось в главе 1, в Azure есть средства, оптимальные для вас и вашей рабочей среды.

Недостаток использования портала, интерфейса командной строки или команд PowerShell — это необходимость нажимать много кнопок в веб-браузере или набирать строки команд для создания среды приложения. Нам помогли бы скрипты, но тогда придется разработать логику одновременного создания нескольких ресурсов или очередность такого создания.

Скрипт, который создает оболочку для команд Azure CLI или PowerShell, — это правильный шаг для создания и развертывания среды приложений не только в Azure, но и на любой другой платформе. Это приближает нас к модели «Инфраструктура как код» (IaC), в которой нет ничего нового для ИТ-специалистов. Эта модель позволяет не полагаться на человека, набирающего команды и выполняющего ряд шагов, а программно создавать инфраструктуру с помощью набора инструкций. В развертывании вручную присутствует человеческий фактор, который часто приводит к незначительным ошибкам и различиям в развернутых виртуальных машинах (см. рис. 6.6).

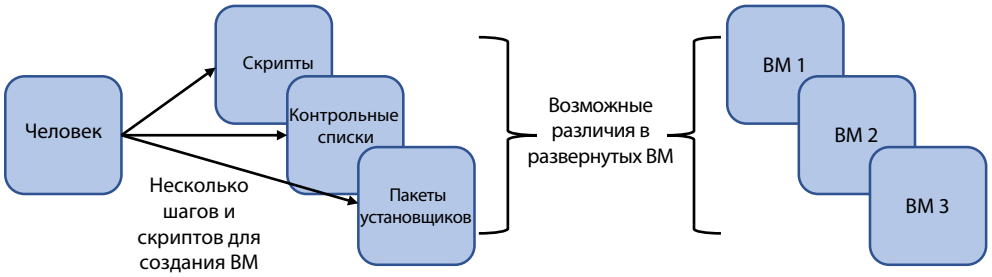


Рис. 6.6. Люди совершают ошибки, например оставляют опечатки в команде или пропускают шаги в развертывании. На выходе можно получить немного разные виртуальные машины. Чтобы исключить оператора из процесса часто используется автоматизация, которая обеспечивает согласованные, идентичные развертывания.

Даже если использовать скрипты, все равно нужно, чтобы кто-то их писал, поддерживал и обновлял по мере выпуска новых версий Azure CLI или модулей PowerShell. Да, возможны серьезные изменения в инструментах для размещения новых функций, хотя они редки.

6.2.1 Создание и использование шаблонов

Шаблоны Resource Manager помогают уменьшить число человеческих ошибок и зависимость от написанных вручную скриптов. Шаблоны пишутся в нотации открытого кроссплатформенного стандарта JavaScript Object Notation (JSON), поэтому их можно редактировать в обычном текстовом редакторе. Шаблоны позволяют создавать согласованные, воспроизводимые развертывания и свести к минимуму число ошибок. Другая встроенная функция шаблонов: платформа понимает зависимости и по возможности создает ресурсы параллельно, уменьшая время развертывания. Если вы создаете 3 виртуальных машины, перед созданием второй ВМ не нужно ждать, пока завершится развертывание первой. Resource Manager может создать все 3 виртуальные машины одновременно.

Пример зависимости — создание виртуального сетевого адаптера, который требуется подключить к подсети. По логике, подсеть должна существовать до создания виртуального сетевого адаптера. В свою очередь, подсеть должна быть частью виртуальной сети, то есть сеть создается раньше подсети. На рис. 6.7 показана цепочка зависимостей в действии. При самостоятельном написании скрипта необходимо тщательно планировать очередность создания ресурсов, но даже тогда вы должны разработать логику, которая сообщит, что родительские ресурсы готовы и можно переходить к зависимым.

Рис. 6.7. Azure Resource Manager обрабатывает зависимости за вас. Платформа знает очередность создания ресурсов и состояние каждого из них без всякой рукописной логики и циклов (которые приходится использовать в собственных скриптах).

Хотите кое-что узнать? Вы уже использовали шаблоны Resource Manager в главе 2 при создании своей первой виртуальной машины. Когда вы создаете виртуальную машину на портале или в Azure CLI, за кадром программно создается и развертывается шаблон. Почему? Зачем изобретать велосипед и создавать всю логику для развертываний? Пусть Azure Resource Manager сделает это за вас!

Посмотрим, как выглядит раздел шаблона Resource Manager. В следующем фрагменте кода показан раздел создания публичного IP-адреса аналогично предыдущим примерам, рассмотренным при создании виртуальной машины.

Фрагмент кода 6.1. Создание публичного IP-адреса в шаблоне Resource Manager

```
{
       "apiVersion": "2019-04-01",
       "type": "Microsoft.Network/publicIPAddresses",
       "name": "publicip",
       "location": "eastus",
       "properties": {
         "publicIPAllocationMethod": "dynamic",
         "dnsSettings": {
           "domainNameLabel": "azuremol"
    }
  }
},
```

Формат JSON понятен даже для незнакомого с ним человека. Сначала определяется тип ресурса (в этом примере — `Microsoft.Network/publicIPAddresses`. Затем вы указываете имя, например `publicip`, и расположение, например `eastus`. Наконец, вы определяете метод выделения (в этом примере — `dynamic`) и метку DNS-имени, например `azuremol`. Это те же параметры, которые вы указывали при использовании портала Azure или интерфейса командной строки. Угадайте, что произойдет, если вы используете PowerShell? Система предложит вам ввести те же параметры.

Шаблоны отличаются тем, что вам вообще не нужно вводить информацию. Она включается в код. Прекрасно, но что делать, если каждый раз требуются разные имена? Как и в скрипте, их можно назначать динамически с помощью параметров и переменных:

- *Параметры* — это запрашиваемые значения. Они часто используются для учетных данных пользователя, имени виртуальной машины и метки DNS-имени.
- *Переменные* — это предварительно назначаемые значения, которые корректируются при каждом развертывании шаблона (например, размер виртуальной машины или имя виртуальной сети.

Попробуйте сейчас

Чтобы просмотреть готовый шаблон Resource Manager, откройте в веб-браузере репозиторий GitHub по адресу http://mng.bz/QyWv.

6.2.2 Создание нескольких однотипных ресурсов

При разработке шаблонов старайтесь заранее продумывать развитие своих приложений в будущем. Возможно, при первом развертывании приложения будет достаточно одной виртуальной машины, но по мере развития приложения могут потребоваться дополнительные экземпляры.

В скриптах традиционного развертывания используется цикл `for` или `while` для создания нескольких однотипных ресурсов. В Resource Manager это встроенная функциональность. Resource Manager предлагает более 50 типов функций, как и большинство языков программирования и сценариев. Часто используемыми функциями Resource Manager являются `length`, `equals`, `or` и `trim`. Функция `copy` позволяет управлять числом создаваемых экземпляров.

Когда используется функция `copy`, Resource Manager создает указанное число ресурсов. При каждом создании ресурса в Resource Manager становится доступным числовое значение для последовательного именования ресурсов. Обратиться к этому значению можно с помощью функции `copyIndex()`. В примере из фрагмента 6.1 создается один публичный IP-адрес. В примере из фрагмента кода 6.2 используется тот же тип поставщика ресурсов `Microsoft.Network/publicIPAddresses`, но создаются 2 публичных IP-адреса. Функция copy определяет число создаваемых адресов, а функция `copyIndex()` последовательно именует их.

Фрагмент кода 6.2. Создание нескольких публичных IP-адресов с помощью функции `copy`

```
{
       "apiVersion": "2019-04-01",
       "type": "Microsoft.Network/publicIPAddresses",
       "name": "[concat('publicip', copyIndex())]",
       "copy": {
         "count": 2
       }
       "location": "eastus",
       "properties": {
```

```
            "publicIPAllocationMethod": "dynamic",
        }
},
```

Кроме того, функция `concat` объединяет название публичного IP-адреса и числовое значение каждого создаваемого экземпляра. После развертывания этого шаблона публичные IP-адреса будут называться `publicip0` и `publicip1`. Не слишком наглядно, но этот базовый пример показывает систему нумерации при создании нескольких ресурсов с помощью функции `copy`.

6.2.3 *Средства для создания собственных шаблонов*

Признаюсь: хотя шаблоны Resource Manager выполнены на совесть и я рекомендую использовать их для создания и развертывания приложений в Azure, вам все равно предстоит писать собственные шаблоны. Существуют средства, упрощающие эту задачу, а также сотни образцов шаблонов от Microsoft и сторонних разработчиков. Один из лучших способов научиться создавать и использовать шаблоны— изучить шаблоны быстрого запуска, которые доступны в репозитории образцов Microsoft по адресу https://github.com/Azure/azure-quickstart-templates.

Если вы хотите, засучив рукава, приступить к написанию собственных шаблонов, я рекомендую вам 2 основных инструмента. Во-первых, это Visual Studio Code — бесплатный кроссплатформенный редактор с открытым исходным кодом (https://code.visualstudio.com). Наряду со встроенными функциями, например системой управления версиями и интеграцией с GitHub, доступны расширения для автоматического создания различных разделов или поставщики ресурсов для создания шаблона (см. рис. 6.8). Чтобы скачать и установить VS Code, выберите «Просмотр» > «Расширения» и выполните поиск в *Azure*.

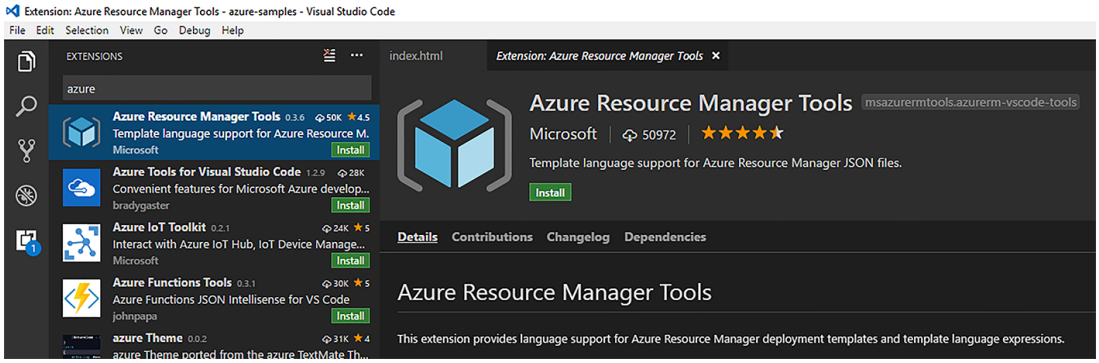


Рис. 6.8. Visual Studio Code предлагает множество расширений, которые улучшают и упрощают создание и использование шаблонов Azure Resource Manager.

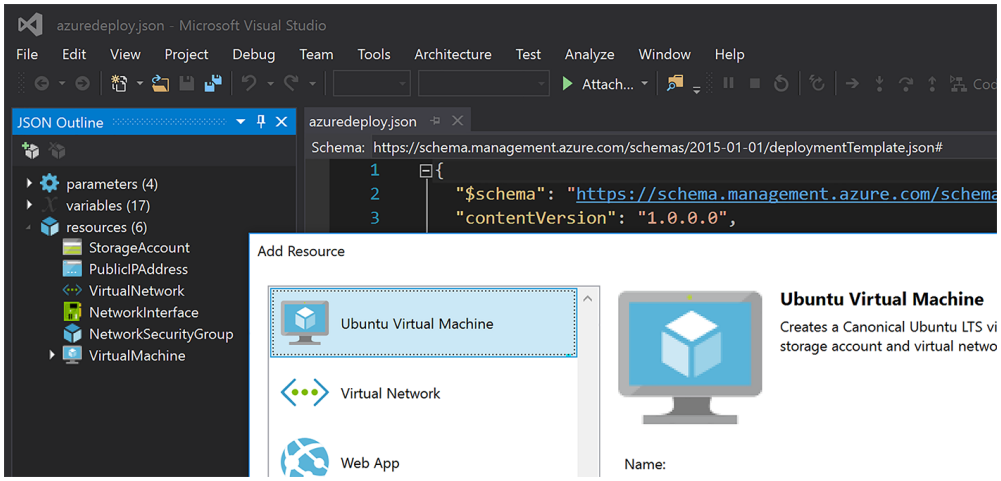


Рис. 6.9. Visual Studio позволяет создавать шаблоны и исследовать ресурсы JSON в графическом режиме.

Графическое средство создания шаблонов Azure Resource Manager — это полнофункциональный редактор Visual Studio (см. рис. 6.9). Существуют версии для Windows и macOS, но каждая требует отдельной лицензии для использования редактора. Доступна версия Community Edition, но будьте осторожны, если собираетесь создавать шаблоны в своей компании. Как правило, требуется лицензированная версия. Обратитесь к специалистам по лицензированию, так как среда Visual Studio предназначена для разработчиков приложений.

Можно, конечно, использовать обычный текстовый редактор. Одна из причин, по которой шаблоны Azure Resource Manager пишутся в формате JSON: для этого не нужны специальные средства. Существует программа обучения работе с JSON, поэтому я рекомендую изучить шаблоны быстрого запуска в репозитории образцов Azure. Обратите внимание на отступы, завершающие запятые и использование круглых, квадратных и фигурных скобок!

Жизнь на Марсе

Существуют средства сторонних производителей и другие способы использования шаблонов в Azure. Компания Hashicorp предоставляет множество инструментов и решений с открытым исходным кодом для облачных вычислений. Одно из них называется Terraform. Terraform позволяет определять все создаваемые ресурсы практически тем же способом, что и собственный шаблон Azure Resource Manager. Кроме того, вы можете определять зависимости и использовать переменные. Разница в том, что технически Terraform — это кроссплатформенный поставщик. Он позволяет использовать одни и те же конструкции и шаблонный подход, например, в Azure, Google Cloud, AWS и vSphere. Разница только в поставщиках ресурсов.

Это на самом деле подход «один шаблон для любого поставщика»? Нет, нисколько. Terraform — это приложение, которое анализирует ваш шаблон и затем связывается с соответствующим поставщиком облачных сервисов, таким как Azure. У вас нет никаких возможностей редактирования, не говоря уже о графических инструментах для создания собственного шаблона. Вы должны выбрать редактор и написать шаблон вручную. Опять же, лучший способ понять Terraform — это изучить документацию и примеры шаблонов.

Я рассказал об этом, потому что Azure предоставляет выбор. Если шаблоны Azure Resource Manager в формате JSON кажутся вам слишком громоздкими, изучите Terraform или подобные продукты. Но не отказывайтесь от развертываний Resource Manager на основе шаблонов. Шаблоны — это наилучшее решение для воспроизводимых, согласованных и масштабируемых развертываний, поэтому найдите оптимальное решение на основе шаблонов, которое подходит именно вам.

6.2.4 Хранение и использование шаблонов

Итак, вам нравятся шаблоны Azure Resource Manager, и вы установили Visual Studio или Code для создания собственных. Как хранить и развернуть их? В практическом упражнении в конце этой главы вам предстоит развернуть шаблон из репозитория образцов Azure на GitHub. Это публичный репозиторий, но можно не показывать свои шаблоны приложений всему миру.

Существует несколько распространенных методов конфиденциального хранения шаблонов Resource Manager:

- Используйте в своей организации частный репозиторий или сетевую папку с файлами.
- Централизованно храните и защищайте шаблоны для развертывания в хранилище Azure.

Не существует правильного или неправильного способа хранения и развертывания шаблонов. Вы можете использовать любые доступные процессы и инструменты. Преимущество репозитория состоит в том, что вы, как правило, получаете некую систему управления версиями, которая позволяет согласованно развернуть и просматривать историю ваших шаблонов. Есть только одно ограничение: при развертывании шаблона необходимо предоставить соответствующие учетные данные для доступа к общему расположению. Подлинность может проверяться по-разному. Например, по имени пользователя или маркеру доступа в URL-адресе шаблона в репозитории или по маркеру подписанного URL-адреса (SAS) , если используется хранилище Azure.

Общедоступные репозитории, например GitHub, также отлично подходят для обучения и совместного использования. Я предлагаю вам хранить шаблоны для рабочей среды конфиденциально, но если вы создаете хороший шаблон для лабораторной среды или тестируете новые функции, размещение на GitHub обеспечит обратную связь с ИТ-сообществом и поможет другим при аналогичных развертываниях. Прежде чем создавать собственные шаблоны, проверьте существующие, чтобы не начинать с нуля и каждый раз не изобретать велосипед.

6.3 *Практическое упражнение: развертывание ресурсов Azure на основе шаблона*

Вся эта теория моделей и подходов хороша, но вы увидите (в идеале) преимущества и эффективность, когда начнете использовать шаблоны на практике.

1. Перейдите к примерам быстрого запуска Azure на портале GitHub (https://github.com/Azure/azure-quickstart-templates) и найдите интересующий вас пример. Лучше всего начать с простой виртуальной машины Linux или Windows.
2. В образцы на GitHub встроены кнопки для развертывания прямо в Azure. Выбрав нужный шаблон, нажмите кнопку «Развернуть в Azure» (см. рис. 6.10) и следуйте инструкциям на портале. Это похоже на то, как вы ранее создавали ВМ в этой книге, за исключением пары подсказок для ввода необходимых параметров. Все остальные ресурсы создаются за вас и абстрагируются.
3. На последнем шаге развертывания примите лицензионное соглашение и нажмите кнопку «Покупка». При развертывании шаблона вы создаете ресурсы Azure, поэтому «Покупка» означает, что вы согласны оплатить их стоимость.

   Один из базовых шаблонов, например простая виртуальная машина Linux или Windows, стоит примерно столько же, сколько любая другая созданная вами ВМ. Убедитесь, что удалили группу ресурсов после завершения развертывания по аналогии с очисткой после любого другого упражнения.

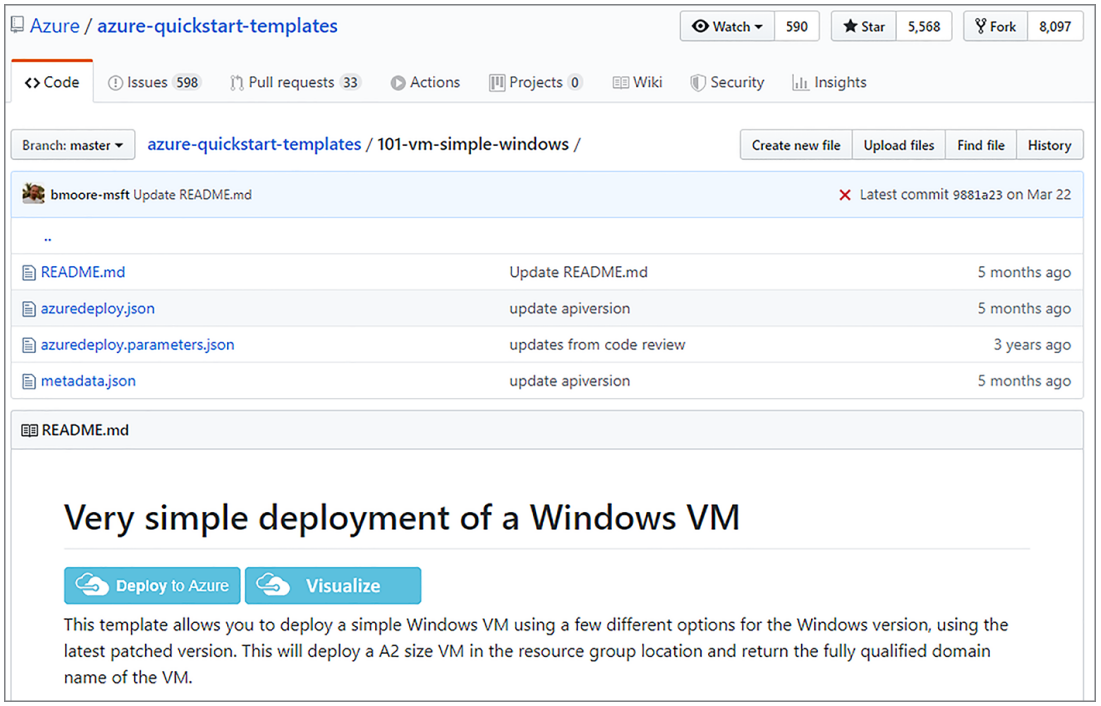


Рис. 6.1. Каждый шаблон Resource Manager в репозитории образцов GitHub содержит кнопку «Развернуть в Azure». При нажатии этой кнопки открывается портал Azure и загружается шаблон. Система предложит вам ввести основные параметры, а оставшуюся часть развертывания выполнит шаблон.

Параметры в шаблонах

Как упоминалось в разделе 6.2.1, в шаблонах можно использовать параметры и переменные. Помните, что параметры — это запрашиваемые значения, а переменные — это динамические значения, которые могут применяться во всем шаблоне. Значения, которые запрашиваются (параметры), варьируются от шаблона к шаблону. Таким образом, в зависимости от выбранного шаблона быстрого запуска возможен запрос одного-двух значений, но может потребоваться ввести семь или восемь.

При разработке шаблонов попытайтесь предусмотреть их повторное использование вами и другими пользователями при развертывании приложений. Вы можете указать стандартное и ограничить допустимые значения. Будьте осторожны со стандартными и допустимыми значениями, иначе вы слишком сильно ограничите пользователей и вынудите их создавать собственные шаблоны. По возможности создавайте повторно используемые базовые шаблоны с достаточной гибкостью.

4 После развертывания шаблона вернитесь в GitHub и проанализируйте файл azure-deploy.json file. Этот файл — шаблон Azure Resource Manager, который использовался для создания и развертывания образца. Убедитесь, что понимаете различные типы ресурсов и применяемые конфигурации. Поработав со многими типами ресурсов и шаблонов Azure, вы намного лучше поймете формат JSON. Честное слово!

7 Высокая доступность и избыточность

Информационные технологии подводили меня поистине бессчетное число раз. Я помню поломку жесткого диска ноутбука за день до конференции, дымящийся блок питания на сервере электронной почты и сбой сетевых интерфейсов на главном маршрутизаторе. Я уже не говорю об обновлениях ОС, драйверов и встроенного ПО! Я уверен, что любой, кто работает в сфере ИТ, с удовольствием поделится ужасными историями из своего опыта. Обычно речь идет о проблемах, возникающих поздно ночью или в критическое для бизнеса время. Хотя существует ли своевременный сбой в удобное время?

Прогнозируя сбои в ИТ-инфраструктуре, вы научитесь планировать и проектировать свои приложения с учетом возможных проблем. В этой главе вы узнаете, как использовать функции высокой доступности и избыточности Azure, чтобы минимизировать перебои в работе, вызванные обновлениями обслуживания и простоями. Эта глава закладывает фундамент для следующих 2–3, в которых вы начнете переход от приложения, работающего на одной виртуальной машине или в веб-приложении, к масштабируемому, глобально распределенному приложению.

7.1 Необходимость избыточности

Чтобы клиенты доверили вам срочный заказ пиццы, они должны знать, что ваши приложения доступны в любое время. Большинство людей не ищут «часы работы» на веб-сайте, особенно если вы работаете в глобальной среде и обслуживаете клиентов по всему миру. Когда они голодны, они хотят есть!

На рис. 7.1 показан базовый пример приложения, работающего на одной виртуальной машине. К сожалению, это приложение создает единую точку отказа. Если виртуальная машина недоступна, значит недоступно и приложение, а клиент остается голодным и недовольным.

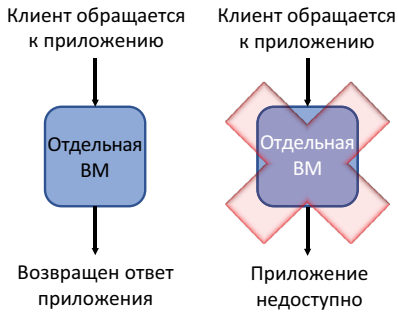


Рис. 7.1. Приложение, работающее на одной ВМ, будет недоступным во время любого ее простоя. В результате клиенты сделают заказы в другом месте или как минимум останутся недовольны вашим обслуживанием.

Если вы едете на машине, скорее всего, у вас есть запасное колесо на случай прокола. Если вы используете ноутбук или планшет, вероятно, вы подключите их к зарядному устройству, если батарея разрядится в процессе работы. В вашем доме или в квартире есть запасные лампочки на случай, если какая-то лампочка сгорит? А как насчет фонарика или свечей на случай отключения электричества?

Большинству людей нравится иметь какой-либо вид избыточности или план резервирования в повседневной жизни и особенно в ИТ-сфере. Если вы готовы к замене автомобильной шины или лампочки, значит продолжительность простоев и сбоев будет минимальной. Проектируя и разрабатывая приложения с учетом избыточности, вы обеспечиваете высокий уровень их доступности для клиентов. Это минимизирует или даже скрывает любые прерывания работы приложения. Все центры обработки данных Azure обеспечивают высокую доступность. Резервные источники питания, несколько сетевых подключений и массивы хранения с запасными дисками — это лишь некоторые из основных концепций избыточности, предоставляемых и управляемых Azure для вас. Но даже вся избыточность от Azure не поможет, если приложение выполняется на одной виртуальной машине. Гибкость и высокая доступность приложений обеспечиваются двумя основными функциями для рабочих нагрузок IaaS:

- *Зона доступности* — позволяет распределять виртуальные машины по физически изолированным сегментам региона Azure, чтобы увеличить избыточность для приложений. Кроме того, зоны обеспечивают высокую доступность сетевых ресурсов: публичных IP-адресов и подсистем балансировки нагрузки.
- *Группа доступности* — позволяет логически группировать виртуальные машины, чтобы распределять их в одном ЦОД Azure и минимизировать перебои в работе, вызванные обновлениями обслуживания или простоями.

Для большинства новых развертываний приложений в Azure я рекомендую использовать зоны доступности. Этот подход обеспечивает гибкость распределения приложения и избыточность для сетевых ресурсов, которые часто определяют, как клиенты в конечном счете обращаются к базовым виртуальным машинам. Чтобы узнать, как работает каждый подход, рассмотрим их подробнее.

7.2 Избыточность инфраструктуры с зонами доступности

Зоны доступности — это физически изолированные центры обработки данных, которые работают с независимыми базовыми системами, например питание и сетевое подключение. Каждый регион Azure, поддерживающий зоны доступности, предоставляет три зоны. Ресурсы создаются в этих зонах и между ними. На рисунке 7.2 показано распределение ресурсов Azure по зонам доступности.

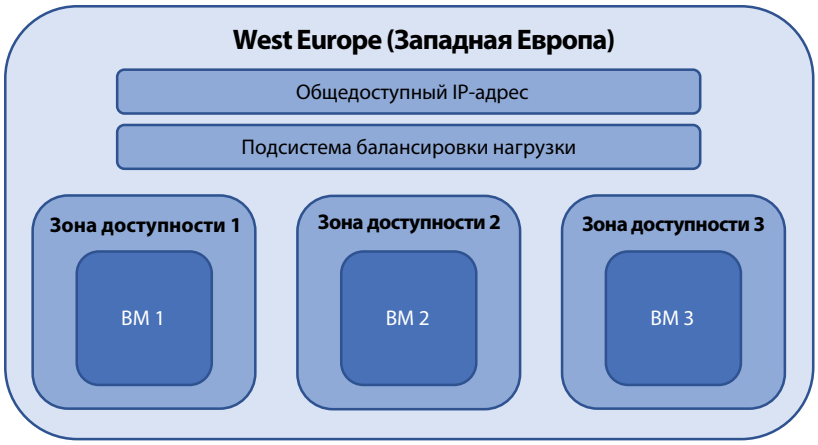


Рис. 7.2. Регион Azure может содержать несколько зон доступности, которые представляют собой физически изолированные ЦОД с независимым питанием, сетью и охлаждением. Ресурсы виртуальной сети Azure, например публичные IP-адреса и подсистемы балансировки нагрузки, могут охватывать все зоны в регионе, что обеспечивает избыточность не только для виртуальных машин.

Благодаря зонам доступности приложения могут выдержать отключение всего центра обработки данных Azure. Конечно, должно произойти серьезное событие, чтобы возникла подобная ситуация, но она все-таки возможна.

При развертывании больших приложений можно создать несколько виртуальных машин в каждой зоне доступности. Несколько виртуальных машин в зоне доступности распределяются по доступному оборудованию в зоне автоматически. Вам ничего не нужно настраивать или контролировать. Даже если обновление обслуживания или сбой оборудования затронет все ВМ внутри зоны, помните, что зоны физически изолированы друг от друга, а значит виртуальные машины в другой зоне по-прежнему будут работать.

Если совсем не повезет, возможно ли одновременное обновление обслуживания моих ВМ в разных зонах? Да, но это маловероятно. Циклы обновлений для зон в регионе смещены. Обслуживание в зонах обновляется по очереди. Зоны доступности обеспечивают высокий уровень абстракции и избыточности, поэтому следует смотреть на все развертывание приложения, а не просто на размещение ВМ в одной зоне.

Включение ресурсов виртуальной сети в зоны доступности намного важнее, чем кажется на первый взгляд. На рисунке 7.3 показано, что произойдет, если ЦОД

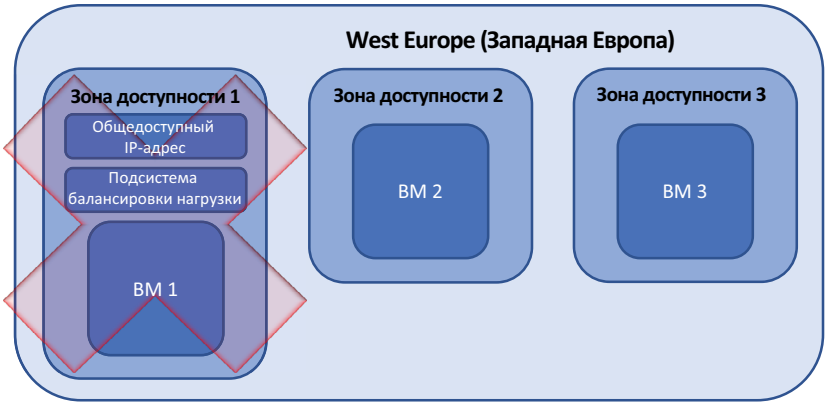


Рис. 7.3. Если сетевые ресурсы подключены к одному центру обработки данных или одной зоне Azure, сбой этого объекта перекроет доступ клиентов к приложению. Неважно, что ВМ в других зонах продолжают работать. Без сетевого подключения, распределяющего трафик от ваших клиентов, все приложение будет недоступно.

станет недоступным для сетевых ресурсов в разных зонах доступности, например для публичного IP-адреса и подсистемы балансировки нагрузки.

В главе 8 мы подробно рассмотрим подсистемы балансировки нагрузки. Сейчас вам достаточно знать, что такая подсистема распределяет трафик по всем доступным ВМ, подключенным к ней. Виртуальные машины сообщают о своей работоспособности через заданные промежутки времени, и подсистема балансировки нагрузки больше не распределяет трафик на ВМ, которая сообщает о своей недоступности. Благодаря подсистеме балансировки нагрузки, работающей в разных зонах доступности, сбой в одном ЦОД Azure приводит к тому, что эти виртуальные машины становятся недоступными и изымаются из ротации при балансировке нагрузки.

Публичный IP-адрес, охватывающий зоны доступности, — это единая точка входа для доступа клиентов к подсистеме балансировки нагрузки и последующего распределения на доступную виртуальную машину. Допустим, публичный IP-адрес развертываемого приложения остается в одном ЦОД Azure. При неполадках в этом центре ни один клиент не сможет перейти по публичному IP-адресу. Клиент не сможет использовать приложение, даже если доступны виртуальные машины для обработки его запроса.

Среди ресурсов, которые могут использовать зоны доступности, как зональные сервисы, так и сервисы, избыточные в пределах зоны:

- *Зональные сервисы* предназначены для таких ресурсов, как виртуальные машины, публичный IP-адрес или балансировщик нагрузки. Весь ресурс сам работает в пределах заданной зоны и может работать автономно, если другая зона недоступна.
- *Сервисы, избыточные в пределах зоны* предназначены для ресурсов, которые могут автоматически реплицироваться между зонами, такими как хранилище и базы данных SQL, избыточные в пределах зоны. Весь ресурс не выполняется в пределах указанной зоны. Его данные распределены по нескольким зонам, чтобы он оставался доступным при возникновении проблемы в одной из зон.

Поддержка зоны доступности доступна для 20 сервисов Azure в 10 регионах. Число сервисов и регионов, интегрированных с зонами доступности, продолжает расти. Однако с учетом ограничений регионов могут быть случаи, когда поддержка зоны доступности недоступна для базовых ресурсов, таких как виртуальные машины. Для этих случаях есть другой тип избыточности ВМ, который можно использовать в любом регионе. Мы рассмотрим его в разделе 7.2.1 «Группы доступности».

7.2.1 Создание сетевых ресурсов в зоне доступности

Чтобы увидеть эти средства обеспечения доступности и избыточности на практике, мы создадим распространенные ресурсы, такие как публичный IP-адрес и балансировщик нагрузки, а затем виртуальные машины. Цель в том, чтобы увидеть, что вам вообще не нужно что-то настраивать, чтобы воспользоваться преимуществами зон доступности в Azure. Это простые примеры, но их можно увидеть в большинстве сред приложений, которые вы будете развертывать.

Публичные IP-адреса и подсистемы балансировки нагрузки можно создавать на одном из 2 уровней: базовом и стандартном. Основное различие — стандартный уровень позволяет сетевому ресурсу использовать зоны доступности. По умолчанию стандартный публичный IP-адрес или подсистема балансировки нагрузки автоматически избыточны в пределах зоны. Дополнительная настройка не предусмотрена. Платформа Azure централизованно хранит метаданные для ресурса в указанном регионе и гарантирует, что ресурс продолжит работу, если одна зона станет недоступной.

Сейчас вам не стоит беспокоиться о том, что происходит с подсистемой балансировки нагрузки и сетевыми ресурсами. Напомню, о чем я говорил в начале: следующие 2–3 главы основаны одна на другой. В главе 8 мы погрузимся в подсистемы балансировки нагрузки, и все прояснится.

Попробуйте сейчас

Чтобы создать сетевые ресурсы, избыточные в зонах доступности, выполните приведенные ниже действия:

1. Щелкните значок Cloud Shell в верхней части панели на портале Azure.
2. Создайте группу ресурсов, например `azuremolchapter7az`:

```
az group create --name azuremolchapter7az --location westeurope
```

3. Создайте стандартный публичный IP-адрес в группе ресурсов. По умолчанию создается *базовый* публичный IP-адрес, который назначается одной зоне. Параметр `--sku standard` указывает Azure создать избыточный межзонный ресурс:

```
az network public-ip create \
  --resource-group azuremolchapter7az \
  --name azpublicip \
  --sku standard
```

4 Создайте подсистему балансировки нагрузки, охватывающую зоны доступности. Опять же, по умолчанию будет создана базовая подсистема балансировки нагрузки, назначаемая одной зоне, которая не обеспечивает высокую доступность, необходимую для приложений. Укажите *стандартный* SKU загрузки для создания подсистемы балансировки нагрузки, избыточной в пределах зоны, как показано ниже:

```
az network lb create \
  --resource-group azuremolchapter7az \
  --name azloadbalancer \
  --public-ip-address azpublicip \
  --sku standard
```

7.2.2 Создание виртуальных машин в зоне доступности

Чтобы создать виртуальную машину в зоне доступности, укажите зону выполнения ВМ. Оптимальный вариант для развертывания нескольких ВМ — создание и использование шаблона. Шаблон определяет и распределяет зоны для каждой виртуальной машины. Если в интернет-магазин пиццы станет обращаться все больше клиентов, вы можете изменить число ВМ в шаблоне и развернуть его повторно. Новые виртуальные машины распределяются по зонам автоматически, поэтому не нужно вручную отслеживать зоны выполнения. В практическом упражнении ВМ автоматически создаются и распределяются по вашему шаблону. Чтобы понять логику указания зоны для виртуальной машины, давайте создадим ВМ и вручную укажем зону.

Попробуйте сейчас

Чтобы создать виртуальную машину в зоне доступности, выполните приведенные ниже действия:

1 На портале Azure в верхней части панели мониторинга щелкните значок Cloud Shell.

2 Создайте виртуальную машину с помощью команды `az vm create`, которую вы использовали в предыдущих главах. Параметр `--zone` указывает зону (1, 2 или 3), в которой будет выполняться виртуальная машина. В следующем примере создается ВМ с именем `zonedvm` в зоне 3:

```
az vm create \
  --resource-group azuremolchapter7az \
  --name zonedvm \
  --image ubuntults \
  --size Standard_B1ms \
  --admin-username azuremol \
  --generate-ssh-keys \
  --zone 3
```

Создание ВМ займет несколько минут. Выходные данные команды указывают зону выполнения ВМ. Эту информацию также можно просмотреть с помощью команды `az vm show`:

```
az vm show \
--resource-group azuremolchapter7az \
```

```
--name zonedvm \
--query zones
```

ПРИМЕЧАНИЕ. ПРИМЕРЫ В УПРАЖНЕНИЯХ «ПОПРОБОВАТЬ» ПРОСТЫ, НО ОНИ ПОКАЗЫВАЮТ, ЧТО ЗОНЫ МОЖНО ИСПОЛЬЗОВАТЬ ПРАКТИЧЕСКИ БЕЗ НАСТРОЙКИ. ВЫ НЕ ИНТЕГРИРОВАЛИ ИЗБЫТОЧНЫЕ В ПРЕДЕЛАХ ЗОНЫ ПОДСИСТЕМУ БАЛАНСИРОВКИ НАГРУЗКИ И ВИРТУАЛЬНУЮ МАШИНУ, НО В ГЛАВЕ 8 МЫ СОЗДАДИМ ПРАКТИЧНУЮ СРЕДУ ПРИЛОЖЕНИЙ, РАСПРЕДЕЛЕННУЮ ПО ЗОНАМ ДОСТУПНОСТИ. ЦЕЛЬ — ПОКАЗАТЬ, ЧТО ИЗБЫТОЧНОСТЬЮ И РАСПРЕДЕЛЕНИЕМ РЕСУРСОВ ЗАНИМАЕТСЯ ПЛАТФОРМА AZURE, ПОЭТОМУ ВЫ МОЖЕТЕ СОСРЕДОТОЧИТЬСЯ НА САМОМ ПРИЛОЖЕНИИ.

7.3 Избыточность для виртуальных машин с помощью групп доступности

Зоны доступности отлично подходят для обеспечения избыточности широкого набора ресурсов, из которых состоят приложения и рабочие нагрузки. Я рекомендую использовать их для новых рабочих нагрузок, где это возможно. Однако иногда необязательно делать все ресурсы избыточными в пределах зоны. Вы также можете создать виртуальные машины в регионе Azure, который сейчас не поддерживает зону доступности.

Чтобы обеспечить избыточность только для виртуальных машин, используйте группы доступности. Они проверены, надежны и доступны во всех регионах. Группы доступности содержат логическую группу виртуальных машин, указывающих платформе Azure тщательно подобранное базовое оборудование, на котором выполняются эти ВМ. Если вы создадите 2 виртуальные машины на одном физическом сервере и на нем произойдет сбой, обе виртуальные машины перестанут работать. Поскольку ЦОД Azure может содержать десятки тысяч (и больше) физических серверов, маловероятно, что обе виртуальные машины окажутся на одном сервере, однако это возможно. Кроме того, причиной может стать не сбой, а обновление обслуживания, которое сделает физический сервер кратковременно недоступным.

А если виртуальные машины работают в одной стойке, подключенной к одному хранилищу или сетевому оборудованию? Это единая точка сбоя, которую мы обсуждали в начале главы.

Группы доступности позволяют платформе Azure создавать виртуальные машины в логических группах, которые называются *доменами сбоя* и *доменами обновления*. Благодаря логическим доменам платформа Azure знает физические границы групп оборудования и равномерно распределяет виртуальные машины между ними. Проблема одного устройства затрагивает только несколько виртуальных машин в группе доступности. Точно так же обновления обслуживания, применяемые к физическому оборудованию, затрагивают только несколько виртуальных машин. Взаимосвязь физического оборудования с логическими доменами сбоя и обновления в группе доступности показана на рис. 7.4.

Зоны доступности выполняют такое же распределение в фоновом режиме, но оно абстрагируется и не раскрывается пользователю. Даже для групп доступности настраивать почти ничего не надо. Однако полезно будет знать, что происходит за кулисами.

7.3.1 Домены сбоя

Домен сбоя — это логическая группа оборудования в центре обработки данных Azure. Он содержит аппаратное обеспечение с питанием или сетевым оборудованием. Вы не контролируете домены сбоя, поэтому вам нечего настраивать на уровне виртуальной машины. Платформа Azure отслеживает домены сбоя, в которых находятся виртуальные машины, и распределяет новые

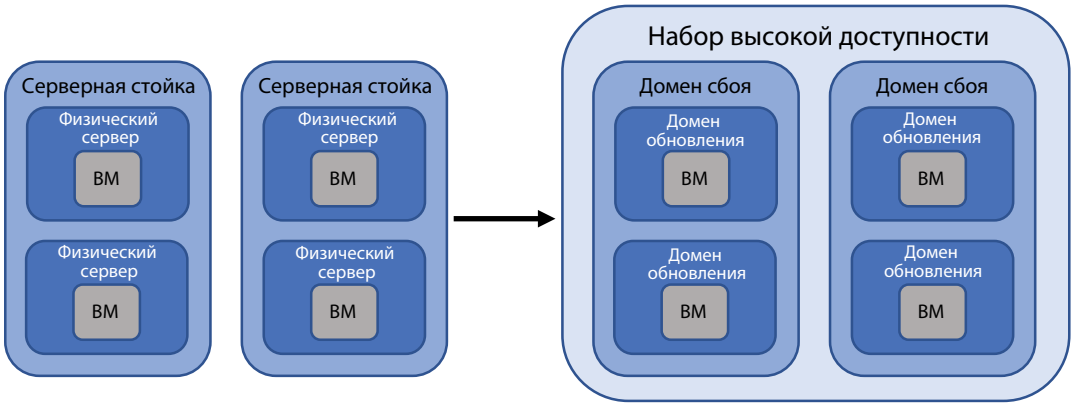


Рис. 7.4. Оборудование в центре обработки данных Azure логически делится на домены обновления и домены сбоя. Благодаря логическим доменам платформа Azure знает, как распределять виртуальные машины по базовому оборудованию в соответствии с требованиями к избыточности. Это базовый пример — скорее всего, домен обновления содержит больше одного физического сервера.

ВМ по этим доменам, обеспечивая их доступность при сбое питания или сетевого коммутатора.

Виртуальные машины, использующие управляемые диски (*все* ваши виртуальные машины должны использовать управляемые диски!), тоже учитывают логические границы и распределение доменов сбоя. Платформа Azure логически назначает кластеры хранилища для доменов сбоя, распределяя виртуальные машины по группам оборудования и управляемые диски между аппаратными средствами хранения данных. Если бы все управляемые диски могли находиться в одном кластере хранилища, не было бы никакого смысла в избыточности для виртуальных машин на серверном оборудовании. И да, управляемые диски также можно использовать с зонами доступности.

7.3.2 Домены обновления

В то время как домены сбоя создают логическую группу оборудования для защиты от сбоев, домены обновления защищают от перебоев при регулярном обслуживании. Для этого домен сбоя логически делится на домены обновления. Опять же, вам здесь нечего настраивать. Благодаря доменам обновления платформа Azure знает, как распределять виртуальные машины в группе доступности.

Инженеры Azure выполняют обслуживание (в основном автоматизированное) и применяют обновления ко всему физическому оборудованию в одном домене обновления, а затем переходят к следующему. Работа по техническому обслуживанию распределяется по доменам обновления. Это гарантирует, что все виртуальные машины в группе доступности не будут одновременно выполняться на обслуживаемом оборудовании. Это такой же процесс, который мы рассматривали в зонах доступности. Распределение ресурсов означает, что невозможна ситуация, в которой все базовое оборудование для ваших ресурсов обновляется одновременно.

В пределах нескольких групп доступности домены не взаимосвязаны. Физические ресурсы в доменах сбоя и обновления для одной группы доступности могут отличаться от ресурсов для другой группы доступности. Это означает, что при создании нескольких групп доступности и распределении виртуальных машин между ними домен сбоя 1, например, не всегда содержит одно и то же физическое оборудование.

7.3.3 Распределение виртуальных машин в группе доступности

Давайте пошагово рассмотрим распределение виртуальных машин по логическим доменам сбоя и обновления, из которых состоит группа доступности. Предположим, есть несколько виртуальных машин, на которых вы можете запустить магазин пиццы, чтобы клиенты не голодали!

Попробуйте сейчас

Чтобы узнать, как работают группы доступности, выполните приведенные ниже действия по развертыванию шаблона Resource Manager:

1. Откройте в веб-браузере шаблон Resource Manager из репозитория образцов на GitHub по адресу https://github.com/fouldsy/azure-mol-samples-2nd-ed/tree/master/ 07/availability-set, а затем нажмите кнопку «Развернуть в Azure». В этом упражнении вы используете шаблон, чтобы быстро развернуть виртуальные машины и изучить их распределение в группе доступности.

   Откроется портал Azure с запросом ряда параметров.

2. Выберите «Создать группу ресурсов» и укажите имя, например `azuremolchapter7`. Выберите регион и укажите данные ключа SSH (их можно получить в Cloud Shell с помощью команды cat ~/.ssh/id_rsa.pub).

   Шаблон создаст группу доступности, содержащую три виртуальные машины. Эти виртуальные машины распределяются по логическим доменам сбоя и обновления. Этот шаблон, созданный на основе того, что вы изучили о Resource Manager в главе 6, использует функцию `copyIndex()` для создания нескольких виртуальных машин и сетевых адаптеров.

3. Чтобы подтвердить создание ресурсов, подробно описанных в шаблоне, установите флажок «Я согласен с вышеуказанными условиями» и нажмите кнопку «Покупка».

Все 3 виртуальные машины в группе доступности будут готовы за несколько минут. Пока идет развертывание, прочтите оставшуюся часть раздела.

При запуске развертывания шаблона создается группа доступности и назначается запрошенное вами число доменов обновления и сбоя. В образце шаблона были определены следующие свойства:

```
"properties": {
    "platformFaultDomainCount": "2",
    "platformUpdateDomainCount": "5",
    "managed": "true"
}
```

Эти свойства создают группу доступности с 2 доменами сбоя и 5 доменами обновления (см. рис. 7.5) и указывают, что ВМ должны использовать управляемые диски, поэтому следует соблюдать соответствующее распределение дисков. Регион, выбранный для группы доступности, определяет максимальное число доменов сбоя и обновления. Регионы поддерживают 2 или 3 домена сбоя и до 20 доменов обновления.

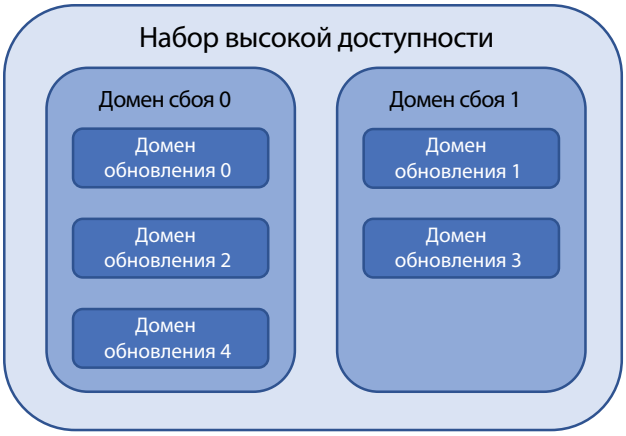


Рис. 7.5. Группа доступности, которую развертывает образец шаблона, содержит два домена сбоя и пять доменов обновления. Используется система нумерации с отсчетом от нуля. Домены обновления создаются последовательно по доменам сбоя.

При создании дополнительных виртуальных машин в группе доступности необходимо учитывать число используемых доменов обновления. Например, 5 доменов обновления означают, что до 20 % виртуальных машин может быть недоступно из-за обслуживания:

- Допустим, что в вашей группе доступности 10 виртуальных машин. Это означает, что 2 из них могут проходить техническое обслуживание одновременно. Если вы хотите разрешить обслуживание только одной виртуальной машины, необходимо создать 10 доменов обновления. Чем больше доменов обновления вы создаете, тем дольше период обслуживания приложения.
- Продолжим предыдущий пример с 10 виртуальными машинами в 10 доменах обновления. Теперь вероятность сбоя в работе приложений существует, пока все 10 из этих доменов обновления не завершат цикл обслуживания. Если у вас только 5 доменов обновления, этот период обслуживания сокращаются. Длительный период обслуживания — это необязательно плохо. Важнее ваша толерантность к потенциальной работе не на максимальной производительности.

Следует помнить, что этими доменами обновления и циклами обслуживания управляет сама платформа Azure. Вам также необходимо учитывать собственные потребности в обновлении и окнах обслуживания.

При создании первой виртуальной машины Azure определяет расположение первой доступной позиции развертывания. Это домен сбоя 0 и домен обновления 0 (см. рис. 7.6).

При создании второй виртуальной машины платформа Azure определяет расположение следующей доступной позиции развертывания. Теперь это домен сбоя 1 и домен обновления 1, как показано на рисунке 7.7.

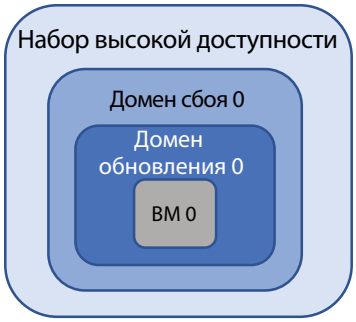


Рис. 7.6. Первая виртуальная машина создается в домене сбоя 0 и домене обновления 0.

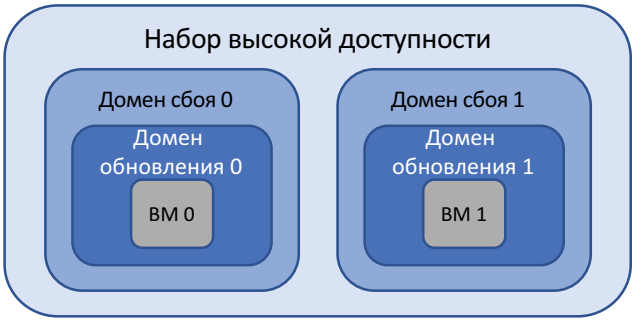


Рис. 7.7. После создания второй виртуальной машины все ВМ равномерно распределяются по доменам сбоя и обновления. Зачастую это считается минимальной избыточностью для защиты приложений.

Поскольку ваш шаблон создает 3 виртуальные машины, как вы думаете, что происходит дальше? Платформа Azure опять определяет расположение следующей доступной позиции развертывания. Вы создали только 2 домена сбоя, поэтому виртуальная машина снова создается в домене сбоя 0, но в другом домене обновления, нежели первая. Третья виртуальная машина создается в домене обновления 2 (см. рис. 7.8).

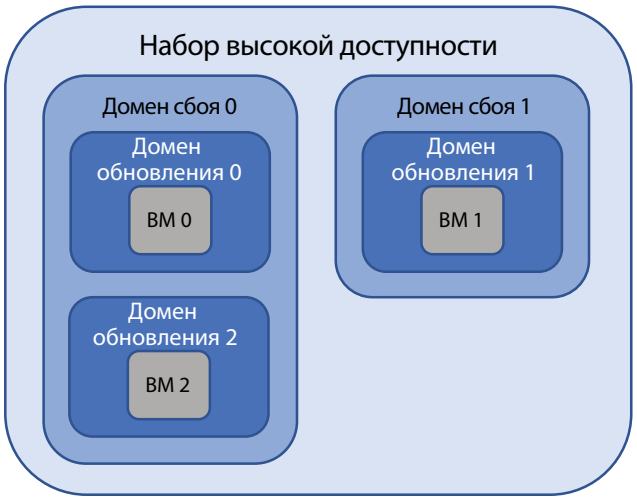


Рис. 7.8. Третья виртуальная машина снова создается в домене сбоя 0, но в домене обновления 2. Несмотря на то, что виртуальные машины 0 и 2 подвержены одинаковому риску сбоя оборудования, они находятся в разных доменах обновления и поэтому не будут проходить регулярное обслуживание одновременно.

Виртуальные машины 0 и 2 находятся в одном домене сбоя, поэтому сбой оборудования может повлиять на обе. Однако регулярное обслуживание затрагивает только одну из них, поскольку они распределены по доменам обновления. Если вы продолжите создавать виртуальные машины, платформа Azure будет распределять их по разным доменами сбоя и обновления. После использования всех 5 доменов обновления шестая виртуальная машина снова создается в домене обновления 0, и цикл продолжается.

7.3.4 *Просмотр распределения виртуальных машин по группам доступности*

Теперь, когда вы освоили теорию распределения виртуальных машин по доменам сбоя и обновления в группе доступности, давайте посмотрим, что произошло с развертыванием шаблона Resource Manager.

Попробуйте сейчас

Чтобы узнать, как распределяются виртуальные машины в группе доступности, выполните приведенные ниже действия:

1. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2. Выберите группу ресурсов, созданную для развертывания шаблона, например `azuremolchapter7`.
3. Выберите группу доступности в списке ресурсов, например `azuremolavailabilityset`.

   В окне «Обзор» представлен список виртуальных машин и связанных с ними доменов сбоя и обновления (см. рис. 7.9).

NAME	STATUS	FAULT DOMAIN	UPDATE DOMAIN
vm0	Running	1	1
vm1	Running	0	2
vm2	Running	0	0

Рис. 7.9. Для группы доступности перечисляются содержащиеся в ней виртуальные машины и отображаются домены сбоя и обновления для каждой ВМ. Эта таблица визуализирует распределение виртуальных машин по логическим доменам.

Если вы наблюдательны, то заметите, что порядок виртуальных машин не соответствует ожидаемому порядку доменов сбоя и обновления. Ошибка ли это? Скорее всего, нет. Изучив пример на рисунке 7.9 и сравнив его с изученным ранее, можно ожидать распределения виртуальных машин в соответствии с таблицей 7.1.

Таблица 7.1. Виртуальные машины группы доступности создаются и распределяются по доменам последовательно

Имя	Домен сбоя	Домен обновления
vm0	0	0
vm1	1	1
vm2	0	2

Что же случилось? Ничего. Вспомните, как Resource Manager создает ресурсы на основе шаблона. Платформа Azure не ждет создания первой виртуальной машины, прежде чем создать вторую. Все три ВМ создаются одновременно. Поэтому время связывания ВМ с группой доступности может отличаться на долю секунды. Порядок не имеет значения, потому что вы не управляете содержимым доменов сбоя и обновления. Это делает платформа Azure. Вам важен факт распределения ВМ, *а не* их *расположение*.

Нет, должны быть красивые номера

Если последовательность важна для вас и *необходимо* распределять виртуальные машины в строгом порядке, укажите Resource Manager создавать виртуальные машины *последовательно*, а не *параллельно*. В этом режиме виртуальные машины создаются одна за другой, поэтому время развертывания увеличивается. Чтобы включить последовательный режим, используйте в своих шаблонах параметр `"mode":"serial"` в функции `copyIndex()`. В результате ВМ будут распределяться строго по очереди.

7.4 Практическое упражнение: развертывание виртуальных машин с высокой доступностью на основе шаблона

Это практическое упражнение позволяет объединить сведения о зонах доступности с материалом по Azure Resource Manager и шаблонам из предыдущей главы, а также закрепить свои знания. Изучите пример шаблона быстрого запуска в этом упражнении, чтобы узнать, как распределить по зонам несколько ВМ с помощью логики и функций. Вы должны не просто развернуть шаблон и идти дальше, а разобраться, как он использует функции из главы 6.

Что такое квота?

В Azure стандартные квоты на подписку не позволят случайно развернуть много ресурсов и забыть о них — это будет стоить слишком дорого. Обычно квоты зависят от типа ресурса и типа подписки и применяются на уровне региона. Полный список квот приведен на странице http://mng.bz/ddcx.

При создании виртуальных машин в следующих главах у вас могут возникнуть проблемы с квотами. Это также может произойти, если вы не удалите ресурсы из предыдущих глав и упражнений. Квоты — это хороший способ отслеживать использование ресурсов. Сообщения об ошибках могут быть непонятными, но если вы видите текст

```
Operation results in exceeding quota limits of Core.
```

```
Maximum allowed: 4, Current in use: 4, Additional requested: 2.
```

значит нужно запросить дополнительны квоты. В Azure нет ничего сложного или необычного. Чтобы просмотреть текущую квоту для данного региона, выполните следующую команду:

```
az vm list-usage --location eastus
```

Если не хватает ресурсов для этого практического упражнения, удалите первые 2 группы ресурсов, созданные в этой главе (`azuremolchapter7` и `azuremolchapter7az`). Если у вас мало стандартных квот, четыре ВМ в этих группах ресурсов могут помешать вам успешно выполнить упражнение.

Чтобы запросить дополнительные квоты для региона, выполните действия, описанные на странице http://mng.bz/Xq2f.

Рассмотрим образец шаблона с несколькими ВМ в зонах доступности и выполним его развертывание.

1. В веб-браузере откройте файл в формате JSON на странице https://github.com/Azure/azure-quick-start-templates/blob/master/201-multi-vm-lb-zones/azuredeploy.json и найдите следующий текст:

   ```
   Microsoft.Compute/virtualMachines
   ```

   Раздел виртуальных машин похож на используемый в главе 6, но обратите внимание на значение свойства zones. Этот раздел объединяет несколько функций шаблона, предназначенных для выбора зоны 1, 2 или 3 при создании виртуальной машины. Таким образом, вам не нужно вручную отслеживать распределение виртуальных машин по зонам и последующее развертывание дополнительных ВМ.
2. В веб-браузере выполните поиск по каждой из следующих строк, чтобы просмотреть разделы для публичного IP-адреса и подсистемы балансировки нагрузки.

   ```
   Microsoft.Network/publicIPAddresses
   Microsoft.Network/loadBalancers
   ```

   Оба ресурса используют стандартный SKU, который по умолчанию обеспечивает избыточность для зоны. Дополнительная настройка не требуется. Давайте посмотрим, как это работает.
3. В веб-браузере откройте шаблон быстрого запуска на странице http://mng.bz/O69a и нажмите кнопку «Развернуть в Azure».
4. Создайте или выберите группу ресурсов, а затем укажите имя пользователя и пароль для виртуальных машин.
5. Введите уникальное DNS-имя, например azuremol.
6. Выберите ОС виртуальных машин: Linux или Windows. Создание виртуальных машин с ОС Windows занимает немного больше времени.

7 Укажите число создаваемых виртуальных машин, например 3.

8 Установите флажок, чтобы согласиться с условиями развертывания шаблона, и нажмите кнопку «Покупка» (см. рис. 7.10).

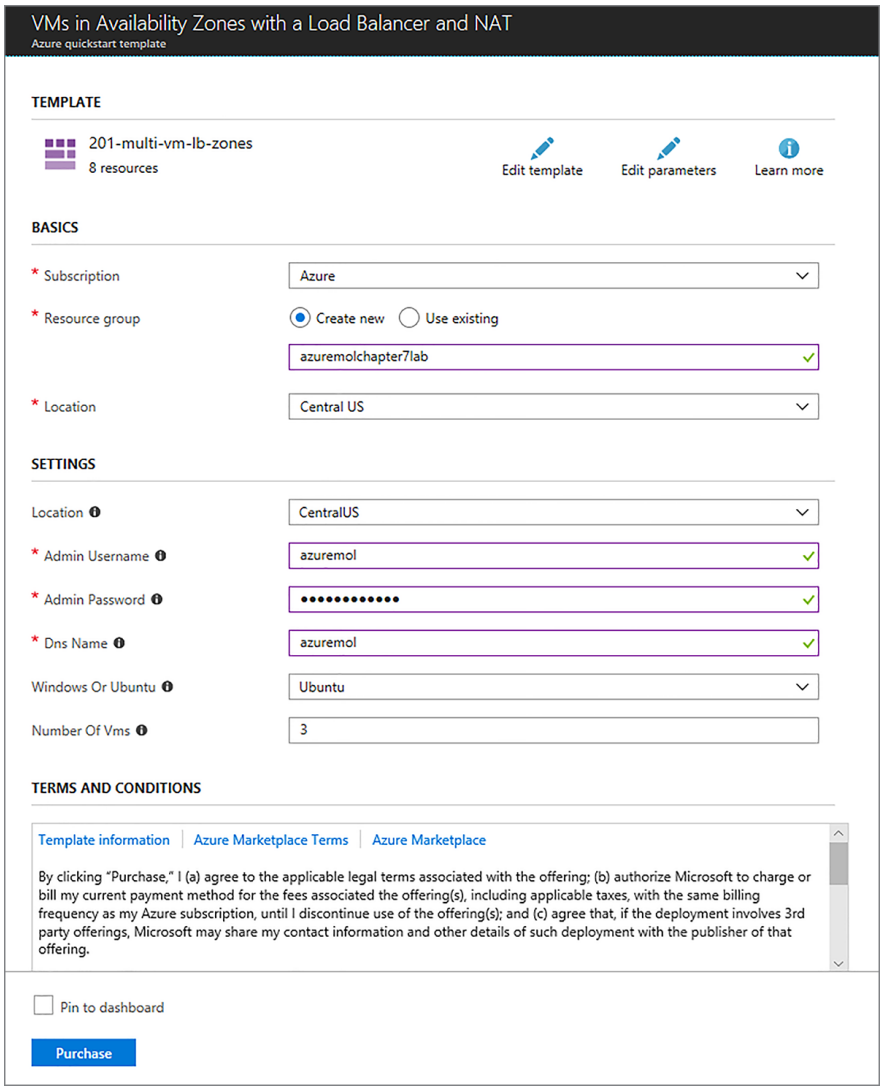


Рис. 7.10. Чтобы развернуть шаблон зоны доступности на портале Azure, укажите группу ресурсов, имя пользователя и пароль, а затем тип ОС и число виртуальных машин, которые необходимо создать. В шаблоне используются циклы, функции `copyIndex()` и `dependsOn`, переменные и параметры, описанные в главе 6.

После создания виртуальных машин используйте портал Azure или команду `az vm show`, чтобы узнать распределение ВМ по зонам. Интересуетесь, как обрабатывает сетевые ресурсы оставшаяся часть шаблона? Тогда глава 8 о подсистемах балансировки нагрузки — именно для вас.

Уборка в проходе 3

Помните, в начале книги я советовал вам убирать за собой, чтобы минимизировать расходы, покрываемые бонусами от Azure? Я настоятельно рекомендую удалить группы ресурсов, созданные в этой главе. В следующих главах мы продолжим создание нескольких виртуальных машин и экземпляров веб-приложений, поэтому не забывайте контролировать затраты и квоты.

При каждом входе на портал Azure вы будете получать всплывающее уведомление о состоянии бонусов Azure. Если ежедневно расходуется значительная сумма, возможно, вы забыли удалить какие-то группы ресурсов.

8 Приложения для балансировки нагрузки

Важный компонент приложений высокой доступности — распределение трафика между всеми виртуальными машинами. Из главы 7 вы узнали, в чем разница между группами и зонами доступности и как создать несколько виртуальных машин в ЦОД или регионах Azure, чтобы обеспечить избыточность приложений. Но высокая доступность и распределение этих ВМ не помогут, если весь трафик клиентов получает только одна виртуальная машина.

Подсистемы балансировки нагрузки — это сетевые ресурсы, которые получают входящий трафик приложения от клиентов, анализируют его для применения фильтров и правил балансировки нагрузки, а затем распределяют запросы по пулу виртуальных машин, на которых выполняется приложение. Azure предлагает разные способы балансировки нагрузки, например разгрузку SSL для больших приложений, использующих зашифрованный сетевой трафик. В этой главе мы рассмотрим различные компоненты подсистемы балансировки нагрузки, настройку правил и фильтров трафика, а также распределение трафика между ВМ. Вы создадите компоненты высокой доступности из главы 8 и подготовитесь к главе 9, посвященной масштабированию ресурсов.

8.1 Компоненты подсистемы балансировки нагрузки в Azure

Подсистемы балансировки нагрузки в Azure работают на 2 уровнях: на уровне 4, где анализируется и распределяется сетевой трафик (на самом деле это транспортный уровень), и на уровне 7, где данные распределяются на основе информации о данных приложения в сетевом трафике. Оба уровня подсистемы балансировки нагрузки работают одинаково (см. рис. 8.1).

Рис. 8.1. Трафик из Интернета поступает в подсистему балансировки нагрузки через публичный IP-адрес, подключенный к пулу внешних IP-адресов. Трафик обрабатывается правилами подсистемы балансировки нагрузки, которые определяют, как и куда он направляется. Зонды работоспособности, подключенные к правилам, гарантируют распределение трафика только по работоспособным узлам. Затем трафик, распределенный правилами подсистемы балансировки нагрузки, поступает во внутренний пул виртуальных сетевых адаптеров, подключенных к ВМ.

Подсистема балансировки нагрузки состоит из нескольких компонентов.

- *Пул внешних IP-адресов* — точка входа в подсистему балансировки нагрузки. Для доступа из Интернета можно подключить общедоступный IP-адрес к пулу внешних IP-адресов. Частные IP-адреса подключаются для внутренних подсистем балансировки нагрузки.
- *Зонды работоспособности* — отслеживают состояние подключенных ВМ. Проверки выполняются регулярно, что гарантирует распределение трафика только по работоспособным ВМ, отвечающим на запросы и правильно реагирующим на трафик.
- *Правила подсистемы балансировки нагрузки* — распределяют трафик по виртуальным машинам. Каждый входящий пакет соотносится с правилами, определяющими входящие протоколы и порты, а затем распределяется в наборе связанных ВМ. Если для входящего трафика нет правил, он удаляется.
- *Правила преобразования сетевых адресов (NAT)* — непосредственно направляют трафик на конкретные виртуальные машины. Например, чтобы предоставить удаленный доступ по SSH или RDP, можно определить правила NAT для перенаправления трафика с внешнего порта на одну ВМ.
- *Пул внутренних IP-адресов* — здесь подключаются виртуальные машины, на которых выполняется приложение. Правила подсистемы балансировки нагрузки связаны с внутренними пулами. Вы можете создавать разные внутренние пулы для разных частей приложений.

Шлюз приложений Azure: расширенная балансировка нагрузки

Подсистемы балансировки нагрузки в Azure работают на уровне сети или на уровне приложения. В этой главе рассматривается обычная подсистема балансировки нагрузки Azure, которая работает на уровне сети (уровень 4 или транспортный протокол). На этом уровне анализируется и распределяется трафик, но у подсистемы балансировки нагрузки нет информации о трафике или запущенных приложениях.

Шлюз приложений Azure — это подсистема балансировки нагрузки, работающая на уровне приложений (уровень 7). Шлюз приложений получает информацию о приложении, выполняющемся на виртуальной машине, что позволяет эффективнее управлять потоками трафика. Основное преимущество шлюза приложений — это возможность обработки зашифрованного веб-трафика HTTPS.

Балансировка нагрузки веб-сайтов с помощью сертификатов SSL позволяет разгрузить процесс, который проверяет и расшифровывает трафик с веб-серверов. На веб-сайтах с большим объемом SSL-трафика проверка и расшифровка может занимать большую часть времени вычислений на виртуальных машинах или в веб-приложениях. Шлюз приложений может проверить и расшифровать трафик, передать чистый веб-запрос на веб-серверы, а затем повторно зашифровать полученный оттуда трафик и вернуть его клиенту.

Шлюз приложений предлагает и другие расширенные функции подсистемы балансировки нагрузки, например возможность передавать трафик на любую конечную точку по IP-адресу, а не только на ВМ Azure. Эти расширенные правила распределения могут быть очень полезны при создании приложений, использующих не только виртуальные машины. Те же базовые концепции применяются в обычной подсистеме балансировки нагрузки. В текущей главе мы рассмотрим, как все это работает в Azure.

8.1.1 Создание пула внешних IP-адресов

В предыдущих главах вы создавали виртуальные машины с публичным IP-адресом, который назначался им непосредственно. Затем вы использовали этот IP-адрес, чтобы получить доступ к ВМ с помощью удаленного подключения (по SSH или RDP) или открыть в браузере веб-сайт, выполняющийся на этой ВМ. При использовании подсистемы балансировки нагрузки вы больше не подключаетесь непосредственно к виртуальным машинам. Теперь, чтобы направить трафик в подсистему балансировки нагрузки и распределить его по виртуальным машинам, необходимо назначить один или несколько IP-адресов внешнему интерфейсу этой подсистемы.

Подсистемы балансировки нагрузки работают в одном из 2 режимов:

- *Подсистема балансировки нагрузки для Интернета* — имеет один или несколько *публичных* IP-адресов, подключенных к пулу внешних IP-адресов. Подсистема балансировки нагрузки для Интернета напрямую получает трафик из Интернета и распределяет его по внутренним виртуальным машинам. Типичный пример — веб-серверы внешнего интерфейса, непосредственно открываемые клиентами в Интернете.
- *Внутренняя подсистема балансировки нагрузки* — имеет один или несколько *частных* IP-адресов, подключенных к пулу внешних IP-адресов. Внутренняя подсистема балансировки нагрузки работает в виртуальной сети Azure, например с виртуальными машинами серверных баз данных. Обычно серверные базы данных или уровни приложений не имеют выхода во внешний мир. Вместо этого набор интерфейсных веб-серверов подключается к внутренней подсистеме балансировки нагрузки, которая распределяет трафик,

не имея прямого открытого доступа. На рисунке 8.2 показано, как внутренняя подсистема балансировки нагрузки распределяет трафик между внутренними ВМ, находящимися вне публичной подсистемы балансировки нагрузки, и ВМ внешнего веб-интерфейса.

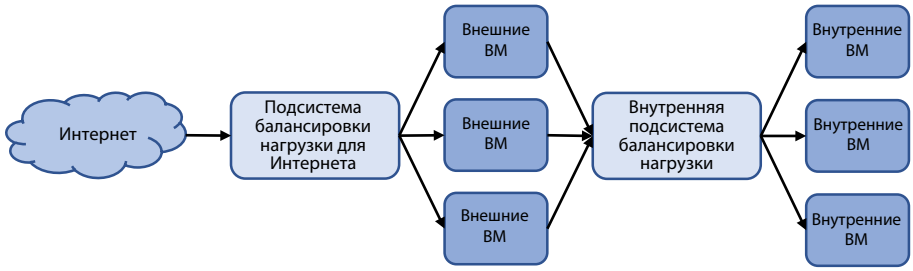


Рис. 8.2. Подсистема балансировки нагрузки для Интернета распределяет трафик между внешними виртуальными машинами, на которых выполняется ваш веб-сайт. Затем этот сайт подключается к внутренней подсистеме балансировки нагрузки, чтобы распределять трафик на уровне базы данных ВМ. Внутренняя подсистема балансировки нагрузки доступна только для внешних ВМ в виртуальной сети Azure.

Режим подсистемы балансировки нагрузки не меняет поведение пула внешних IP-адресов. Вы назначаете один или несколько IP-адресов, которые используются при запросе доступа к подсистеме балансировки нагрузки. Для пула внешних IP-адресов можно настроить адреса IPv4 и IPv6. Это позволяет настраивать прямые связи IPv6 между клиентами и виртуальными машинами как потоки трафика в подсистему балансировки нагрузки и из нее.

Попробуйте сейчас

Чтобы понять взаимосвязь между компонентами подсистемы балансировки нагрузки, создайте подсистему и пул внешних IP-адресов по приведенным ниже инструкциям:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Создайте группу ресурсов с помощью команды `az group create`.

   Укажите имя группы ресурсов, например `azuremolchapter8`, и расположение:

   ```
   az group create --name azuremolchapter8 --location westeurope
   ```

   Если вы хотите использовать зоны доступности, рассмотренные в главе 7, выберите поддерживающий их регион.
3. Создайте публичный IP-адрес с помощью **команды** `az network public-ip create`.

   В главе 7 говорилось, что зоны доступности обеспечивают избыточность для сетевых ресурсов, поэтому создайте стандартный, избыточный в пределах зоны публичный IP-адрес, а также укажите имя, например `publicip`:

   ```
   az network public-ip create \
     --resource-group azuremolchapter8 \
     --name publicip \
     --sku standard
   ```

Чтобы создать публичный IPv6-адрес, добавьте `--version IPv6` в предыдущую команду. Для этих упражнений можно использовать IPv4-адреса.

4 Создайте подсистему балансировки нагрузки и назначьте публичный IP-адрес пулу внешних IP-адресов. Чтобы добавить публичный IP-адрес, укажите параметр `--public-ip-address`. При создании внутренней подсистемы балансировки нагрузки используйте параметр `--private-ip-address`.

По аналогии с публичным IP-адресом создайте стандартную, избыточную в пределах зоны, подсистему балансировки нагрузки для всех зон доступности.

```
az network lb create \
  --resource-group azuremolchapter8 \
  --name loadbalancer \
  --public-ip-address publicip \
  --frontend-ip-name frontendpool \
  --backend-pool-name backendpool \
  --sku standard
```

Через несколько страниц мы приступим к рассмотрению внутреннего пула.

8.1.2 Создание и настройка зондов работоспособности

Если на одной из виртуальных машин, на которых выполняется приложение, возникает неполадка, должна ли подсистема балансировки нагрузки по-прежнему распределять трафик на эту ВМ? Возможно, клиент, обратившийся в ваш магазин пиццы, будет направлен на эту виртуальную машину и не сможет заказать еду. Подсистема балансировки нагрузки отслеживает состояние виртуальных машин и удаляет проблемные. Подсистема балансировки нагрузки продолжает отслеживать работоспособность и возвращает виртуальную машину в пул для распределения трафика, если та снова отвечает корректно.

Зонд работоспособности может работать в следующих режимах:

- *На основе порта* — подсистема балансировки нагрузки проверяет ответ виртуальной машины, поступающий на конкретный порт определенного протокола, например TCP-порт 80. Пока виртуальная машина отвечает зонду работоспособности на TCP-порте 80, она остается в системе распределении трафика. В противном случае подсистема балансировки нагрузки удаляет ВМ из распределения трафика (см. рис. 8.3). Этот режим не гарантирует, что виртуальная машина правильно распределит трафик, а лишь определяет, что сервис сетевого подключения и назначения возвращает ответ.
- *На основе HTTP-пути* — создается настраиваемая страница, например health.html, которая размещается на каждой виртуальной машине. С помощью настраиваемого зонда работоспособности можно проверять доступ к хранилищу образов или подключение к базе данных. В этом режиме подсистема балансировки нагрузки распределяет трафик на ВМ, если страница проверки работоспособности возвращает HTTP-код 200 (см. рис. 8.4). При использовании зонда работоспособности на основе порта возможна ситуация, когда текущий веб-сервер работает, но не подключен к базе данных. При использовании настраиваемой страницы подсистема балансировки нагрузки подтверждает, что ВМ способна обработать трафик клиента.

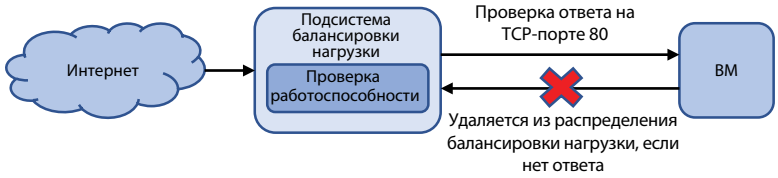


Рис. 8.3. Зонд работоспособности на основе порта, выполняемый подсистемой балансировки нагрузки, проверяет наличие ответа виртуальной машины на определенном порту конкретного протокола. Если виртуальная машина не отвечает в течение заданного периода, подсистема балансировки нагрузки удаляет ее из распределения трафика. Если виртуальная машина снова отвечает правильно, зонд работоспособности определяет это и возвращает машину в распределение трафика подсистемой балансировки нагрузки.

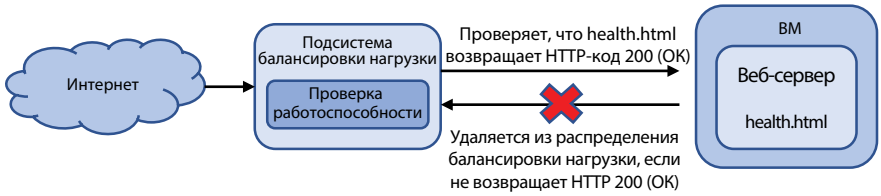


Рис. 8.4. Виртуальная машина, которая запускает веб-сервер и имеет настраиваемую страницу health.html, остается в распределении трафика при условии, что зонд работоспособности получает HTTP-код 200 (OK). Если процесс веб-сервера не может вернуть запрошенные страницы, подсистема балансировки нагрузки удаляет их из распределения трафика. Это позволяет тщательнее проверять состояние веб-сервера по сравнению с зондами работоспособности на основе порта.

Создание настраиваемой страницы для проверки работоспособности требует дополнительных усилий, но улучшение качества обслуживания клиентов того стоит. Страница проверки работоспособности не должна быть сложной. Это может быть обычная HTML-страница, подтверждающая, что веб-сервер в состоянии обрабатывать страницы. Если страница проверки работоспособности отсутствует, а веб-сервер работает некорректно, ВМ остается доступной на TCP-порте 80, поэтому зонд работоспособности на основе порта считает ее работоспособной. Зонд работоспособности на основе HTTP-пути требует, чтобы веб-сервер корректно возвращал HTTP-ответ. При зависании или сбое процесса веб-сервера HTTP-ответ не отправляется, поэтому ВМ удаляется из распределения трафика.

Частота, с которой зонд работоспособности проверяет виртуальную машину, и ответ настраиваются с помощью следующих параметров:

- *Интервал* — определяет, как часто зонд работоспособности проверяет состояние виртуальной машины. Стандартный зонд проверяет состояние каждые 15 секунд.
- *Пороговое значение* — определяет число последовательных сбоев ответа зонду работоспособности, после которого подсистема балансировки нагрузки удаляет ВМ из распределения трафика. Стандартный зонд работоспособности допускает два последовательных сбоя до удаления ВМ из распределения трафика.

Попробуйте сейчас

Чтобы создать зонд работоспособности для подсистемы балансировки нагрузки (см. рис. 8.4), выполните действия, приведенные ниже:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Укажите имя зонда работоспособности, например `healthprobe`. Чтобы настроить зонд работоспособности для веб-сервера, укажите HTTP-порт 80 и настраиваемую страницу проверки работоспособности health.html. В разделе 8.2 вы создадите эту страницу проверки работоспособности на виртуальных машинах. Чтобы узнать, как настраивается ответ зонда работоспособности, укажите 10-секундный интервал и 3 последовательных сбоя для порогового значения:

```
az network lb probe create \
  --resource-group azuremolchapter8 \
  --lb-name loadbalancer \
  --name healthprobe \
  --protocol http \
  --port 80 \
  --path health.html \
  --interval 10 \
  --threshold 3
```

Как добиться, чтобы созданный зонд работоспособности проверял состояние виртуальных машин? Зонды работоспособности связаны с правилами подсистемы балансировки нагрузки. Один и тот же зонд работоспособности можно использовать с несколькими правилами подсистемы балансировки нагрузки. Помните главу 5, где вы создали группы безопасности сети (NSG) и правила? Группы безопасности сети могут быть связаны с несколькими виртуальными машинами или подсетями виртуальной сети. Аналогичное отношение «один ко многим» применяется к зондам работоспособности.

Давайте запустим зонд работоспособности, для чего создадим правила подсистемы балансировки нагрузки.

8.1.3 Определение распределения трафика с помощью правил подсистемы балансировки нагрузки

Распределяя трафик на внутренние ВМ через подсистему балансировки нагрузки, можно определить условия перенаправления пользователя на одну и ту же виртуальную машину. Вы можете сохранять соединение пользователя с одной и той же ВМ в течение одного сеанса или позволить ему возвращаться, обеспечив сходство виртуальных машин на основе исходного IP-адреса. На рисунке 8.5 показан пример стандартного режима сходства сеансов.

В режиме сходства сеансов поток трафика обрабатывается хэшем с 5 кортежами (исходный IP-адрес, исходный порт, целевой IP-адрес, конечный порт и тип протокола). В течение сеанса каждый запрос пользователя к веб-серверу через TCP-порт 80 направляется на одну и ту же внутреннюю машину.

Что происходит, когда клиент закрывает сеанс в браузере? При следующем подключении запускается новый сеанс. Поскольку подсистема балансировки нагрузки распределяет трафик между всеми работоспособными ВМ в пуле внутренних IP-адресов, пользователь может снова подключиться к той же виртуальной машине, но чем больше ВМ в пуле, тем выше вероятность, что пользователь подключится к другой.

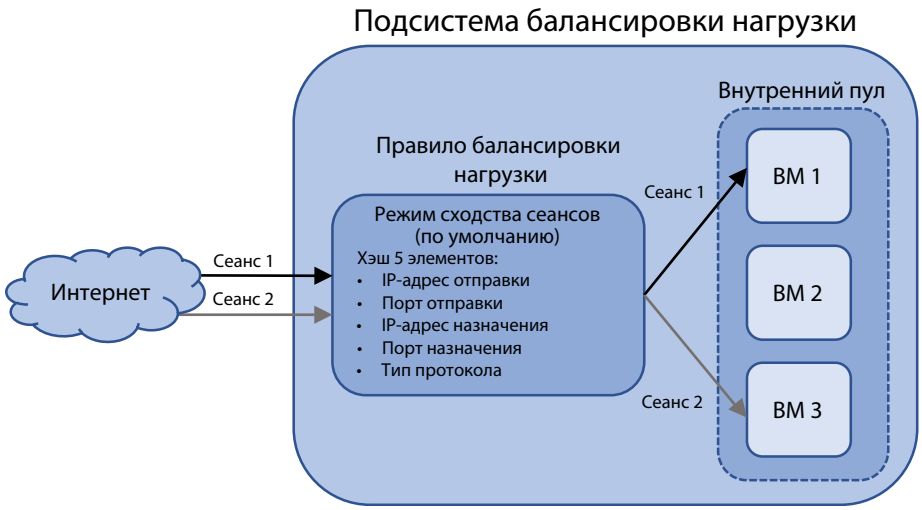


Рис. 8.5. В режиме сходства сеансов пользователь подключается к одной и той же внутренней виртуальной машине только на время своего сеанса.

Вам как владельцу и разработчику приложения может потребоваться, чтобы пользователь подключался к той же ВМ, что и в предыдущем сеансе. Если, например, приложение передает файлы или использует UDP вместо TCP, скорее всего, потребуется, чтобы обработку запросов продолжила та же ВМ. Для подобных сценариев можно настроить режим сходства исходного IP-адреса в правилах подсистемы балансировки нагрузки. На рисунке 8.6 показан пример режима сходства исходного IP-адреса.

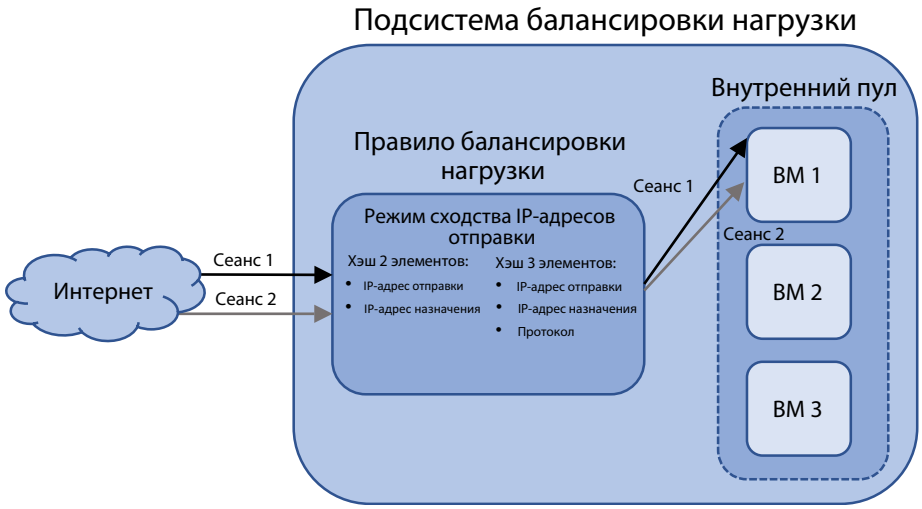


Рис. 8.6. Если вы настроили правила подсистемы балансировки нагрузки для использования режима сходства исходного IP-адреса, пользователь может закрыть и снова запустить сеанс, сохраняя подключение к той же внутренней ВМ. Режим сходства исходного IP-адреса может использовать хэш из 2 кортежей (исходный и целевой IP-адрес) или хэш из 3 кортежей (дополненный протоколом).

Попробуйте сейчас

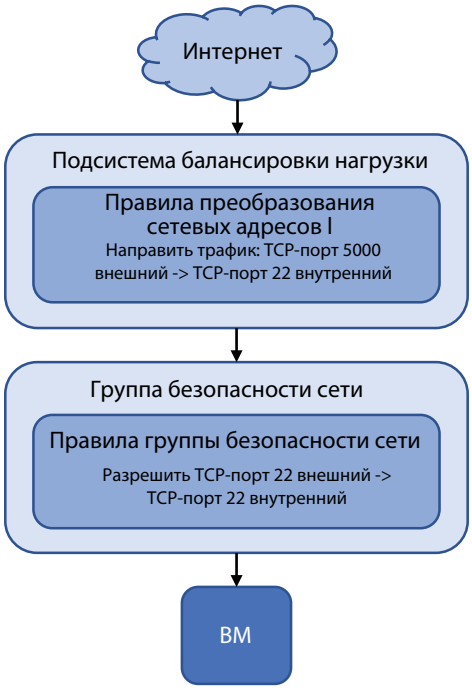
Чтобы создать правило подсистемы балансировки нагрузки, использующее зонд работоспособности, выполните приведенные ниже действия:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Укажите имя правила подсистемы балансировки нагрузки, например `httprule`.
3. Укажите внешний порт, на который поступает трафик, и внутренний порт для распределения трафика. В следующем базовом примере трафик поступает на порт 80, а затем распределяется на порту 80:

```
az network lb rule create \
  --resource-group azuremolchapter8 \
  --lb-name loadbalancer \
  --name httprule \
  --protocol tcp \
  --frontend-port 80 \
  --backend-port 80 \
  --frontend-ip-name frontendpool \
  --backend-pool-name backendpool \
  --probe-name healthprobe
```

На виртуальной машине, которая выполняет несколько сайтов и отвечает на разных портах, это правило может направить трафик на конкретный веб-сайт на этой ВМ.

8.1.4 *Маршрутизация прямого трафика с правилами преобразования сетевых адресов*



Правила подсистемы балансировки нагрузки распределяют трафик между внутренними пулами виртуальных машин, поэтому нет никакой гарантии, что вы сможете подключиться к конкретной ВМ для обслуживания или управления. Как подключиться к определенной виртуальной машине, которая находится вне подсистемы балансировки нагрузки? Завершим изучение подсистем балансировки нагрузки, рассмотрев правила преобразования сетевых адресов (NAT), которые позволяют управлять потоком определенного трафика, направляя его на одну ВМ. На рисунке 8.7 показано перенаправление конкретного трафика на отдельные виртуальные машины с помощью правил NAT.

Рис. 8.7. Трафик в подсистеме балансировки нагрузки обрабатывается правилами NAT. Если протокол и порт соответствуют правилу, трафик перенаправляется на определенную внутреннюю виртуальную машину. Зонды работоспособности не подключены, поэтому подсистема балансировки нагрузки не проверяет работоспособность ВМ, прежде чем перенаправить трафик. Трафик покидает подсистему балансировки нагрузки и обрабатывается правилами NSG. Разрешенный трафик передается на виртуальную машину.

Правила NAT работают вместе с правилами NSG. Виртуальная машина может получать трафик, только если существует правило NSG, которое разрешает тот же трафик, что и правило NAT подсистемы балансировки нагрузки.

Зачем создавать правила NAT? Что делать, если нужно подключиться к определенной ВМ (и вы не используете Бастион Azure, упомянутый в главе 2) по SSH или RDP либо к внутреннему серверу баз данных с помощью средств управления? Если подсистема балансировки нагрузки распределяет трафик между внутренними виртуальными машинами, вы будете тщетно пытаться подключиться к нужной ВМ.

Помните о безопасности

Мы рассмотрим некоторые вопросы безопасности в 3-ей части этой книги, но это неизменно актуальный вопрос при создании и выполнении приложений в Azure. Безопасность нельзя добавить потом. С ростом использования облачных вычислений и одноразовых ВМ и веб-приложений можно легко нарушить основные рекомендации по безопасности. Если вы работаете в Azure в рамках корпоративной подписки, убедитесь, что создаете ресурсы, которые не предоставляют злоумышленникам даже случайный доступ к вашей инфраструктуре.

Что представляет опасность? Даже кое-что из уже проделанного в этой книге. Порты удаленного управления по SSH и RDP нельзя открывать для публичного Интернета, как вы это сделали. По крайней мере, следует запретить доступ к ним с определенных IP-адресов.

Рекомендуется использовать управляемый сервис, такой как Бастион Azure, или вручную создать одну защищенную ВМ с доступными инструментами удаленного управления. При необходимости вы подключаетесь к хосту Бастиона Azure или к одной защищенной ВМ, а затем через внутреннюю виртуальную сеть Azure — к дополнительным виртуальным машинам. Вы применили базовый подход с использованием виртуальной машины узла-бастиона в главе 5. Такой подход сводит к минимуму интенсивность атак и уменьшает необходимость в правилах NSG и правилах NAT подсистемы балансировки нагрузки. В главе 16 обсуждается центр безопасности Azure и возможность динамически запрашивать и открывать порты удаленного управления на оптимальный определенный период.

Даже если вы используете частную подписку Azure, которая не связана с другими подписками Azure в школе или на работе, постарайтесь минимизировать число удаленных подключений.

Попробуйте сейчас

Чтобы создать правило балансировки нагрузки NAT, выполните приведенные ниже действия:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Укажите имя правила NAT подсистемы балансировки нагрузки, например `natrulessh`, и используемый пул внешних IP-адресов. Правило NAT проверяет трафик на соответствие заданному протоколу и порту, например TCP-порт 50001. При совпадении трафик перенаправляется на внутренний порт 22:

```
az network lb inbound-nat-rule create \
  --resource-group azuremolchapter8 \
  --lb-name loadbalancer \
  --name natrulessh \
  --protocol tcp \
  --frontend-port 50001 \
  --backend-port 22 \
  --frontend-ip-name frontendpool
```

На данный момент вы создали базовую подсистему балансировки нагрузки. Проверьте компоненты подсистемы балансировки нагрузки:

```
az network lb show \
  --resource-group azuremolchapter8 \
  --name loadbalancer
```

Пулу внешних IP-адресов назначен публичный IP-адрес, и вы создали зонд работоспособности, чтобы проверять состояние настраиваемой страницы работоспособности веб-сервера. Создано правило балансировки нагрузки для распределения веб-трафика от клиентов во внутренний пул, которое использует зонд работоспособности. У вас также есть правило NAT балансировщика нагрузки, которое разрешает трафик SSH, но пока нет ВМ для получения этого трафика. Клиенты вашего магазина пиццы голодны, поэтому давайте создадим несколько ВМ. На них будет выполняться ваше веб-приложение и распределяться трафик от подсистемы балансировки нагрузки.

8.1.5 Назначение групп ВМ для внутренних пулов

В последнем разделе подсистемы балансировки нагрузки определяются внутренние пулы с одной или несколькими ВМ. Эти виртуальные машины выполняют одни и те же компоненты приложений. Таким образом, подсистема балансировки нагрузки распределяет трафик по данному внутреннему пулу — предполагается, что любая ВМ в пуле правильно ответит на запрос клиента. На рисунке 8.8 объясняется, как серверные пулы логически группируют ВМ, которые выполняют одни и те же приложения.

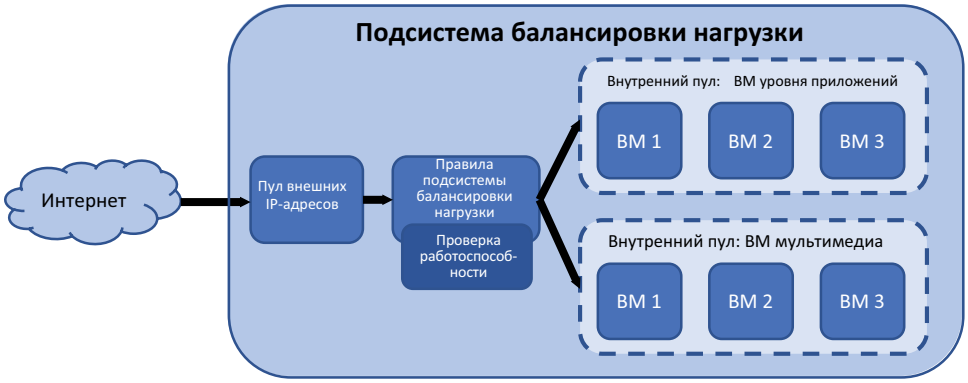


Рис. 8.8. В подсистеме балансировки нагрузки можно создать один или несколько внутренних пулов. Каждый внутренний пул содержит одну или несколько виртуальных машин, которые выполняют один компонент приложения. В этом примере один внутренний пул содержит ВМ уровня веб-приложений, а другой — ВМ для мультимедиа, например изображений и видео.

Вы создаете и используете подсистему балансировки нагрузки для ВМ, но все работает на уровне виртуальной сети. Пул внешних IP-адресов использует общедоступные или частные IP-адреса. Зонд работоспособности просматривает ответы по заданному порту или протоколу. Даже если используется зонд HTTP, подсистема балансировки нагрузки ищет положительный сетевой ответ. Правила балансировки нагрузки распределяют трафик с внешнего порта во внешнем пуле на порт во внутреннем пуле.

При назначении виртуальных машин для внутреннего пула, который получает трафик от подсистемы балансировки нагрузки, к ней подключается виртуальный сетевой адаптер. ВМ подключается к виртуальному сетевому адаптеру. Согласно главе 5, разделение виртуальных машин и виртуального сетевого адаптера улучшает управление ресурсами. Правила NSG определяют, какой трафик разрешено передавать на виртуальную машину, но они применяются к подсети виртуальной сети или к виртуальному сетевому адаптеру, а не к ВМ.

Что это означает для настройки пулов внутренних IP-адресов? Перед подключением ВМ к подсистеме балансировки нагрузки необходимо создать остальные ресурсы виртуальной сети. Несколько глав назад вы изучали создание сетевых ресурсов. Давайте проверим, что вы запомнили.

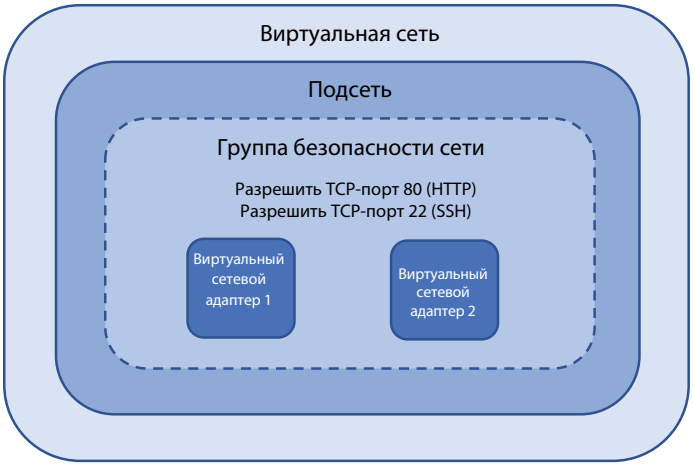


Рис. 8.9. В этом упражнении создаются сеть, подсеть и виртуальные сетевые адаптеры, защищенные с помощью NSG. Правила, подключенные к NSG, разрешают трафик HTTP и SSH.

Попробуйте сейчас

Чтобы создать дополнительные сетевые ресурсы (см. рис. 8.9), выполните действия, приведенные ниже:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Создайте виртуальную сеть и подсеть:

```
az network vnet create \
  --resource-group azuremolchapter8 \
  --name vnetmol \
  --address-prefixes 10.0.0.0/16 \
  --subnet-name subnetmol \
  --subnet-prefix 10.0.1.0/24
```

Скорее всего, эти сетевые ресурсы уже существуют. Кроме того, возможно повторение имен и диапазонов IP-адресов, которые вы использовали в главе 5. Чтобы не возникала проблема повторного использования диапазонов IP-адресов, необходимо очищать ресурсы Azure в конце каждой главы. Просто помните, что, как правило, не нужно создавать виртуальную сеть и подсеть при каждом создании подсистемы балансировки нагрузки. Вы можете использовать существующие ресурсы виртуальной сети.

3 Создайте группу безопасности сети (NSG):

```
az network nsg create \
  --resource-group azuremolchapter8 \
  --name webnsg
```

4 Создайте правило NSG, разрешающее трафик с TCP-порта 80 на ваши виртуальные машины. Это правило необходимо, чтобы ВМ веб-сервера могли получать трафик клиентов и отвечать на него:

```
az network nsg rule create \
  --resource-group azuremolchapter8 \
  --nsg-name webnsg \
  --name allowhttp \
  --priority 100 \
  --protocol tcp \
  --destination-port-range 80 \
  --access allow
```

5 Добавьте еще одно правило, разрешающее трафик SSH для удаленного управления. Это правило NSG работает с правилом NAT подсистемы балансировки нагрузки, созданным в разделе 8.1.4 для одной из ВМ:

```
az network nsg rule create \
  --resource-group azuremolchapter8 \
  --nsg-name webnsg \
  --name allowssh \
  --priority 101 \
  --protocol tcp \
  --destination-port-range 22 \
  --access allow
```

6 Свяжите NSG с подсетью, созданной на шаге 2. Правила NSG применяются ко всем виртуальным машинам, подключаемым к этой подсети:

```
az network vnet subnet update \
  --resource-group azuremolchapter8 \
  --vnet-name vnetmol \
  --name subnetmol \
  --network-security-group webnsg
```

7 Подсистема балансировки нагрузки работает с виртуальными сетевыми адаптерами, поэтому создайте два виртуальных сетевых адаптера и подключите их к подсети виртуальной сети. Кроме того, укажите имя подсистемы балансировки нагрузки и пул внутренних адресов, к которому подключаются виртуальные сетевые адаптеры. Правило NAT подсистемы балансировки нагрузки подключается только к первому виртуальному сетевому адаптеру:

```
az network nic create \
  --resource-group azuremolchapter8 \
  --name webnic1 \
  --vnet-name vnetmol \
  --subnet subnetmol \
  --lb-name loadbalancer \
  --lb-address-pools backendpool \
  --lb-inbound-nat-rules natrulessh
```

8 Аналогичным образом создайте второй виртуальный сетевой адаптер, исключая правило NAT подсистемы балансировки нагрузки:

```
az network nic create \
  --resource-group azuremolchapter8 \
  --name webnic2 \
  --vnet-name vnetmol \
  --subnet subnetmol \
  --lb-name loadbalancer \
  --lb-address-pools backendpool
```

8.2 Создание и настройка виртуальных машин с использованием подсистем балансировки нагрузки

Сделаем паузу и оценим ваше творение. На рисунке 8.10 показана общая картина ваших сетевых ресурсов и подсистемы балансировки нагрузки. Обратите внимание, как интегрированы ресурсы. Подсистема балансировки нагрузки не может существовать сама по себе. Виртуальные сетевые адаптеры должны подключаться

Рис. 8.10. ВМ на этом этапе не создавались, поскольку конфигурация подсистемы балансировки нагрузки относится только к ресурсам виртуальной сети. Подсистема балансировки нагрузки тесно связана с ресурсами виртуальной сети.

к подсистеме балансировки нагрузки для распределения трафика. Для виртуальных сетевых адаптеров требуются виртуальная сеть и подсеть, в идеале защищенные с помощью NSG. Виртуальные машины, на которых будет выполняться ваше приложение, практически не участвуют в создании и настройке подсистемы балансировки нагрузки.

Вы создали много сетевых ресурсов и настроили несколько компонентов подсистемы балансировки нагрузки. Общедоступный IP-адрес и подсистема балансировки нагрузки были созданы как ресурсы, избыточные в пределах зоны доступности, поэтому давайте создадим две ВМ в разных зонах, чтобы подчеркнуть, как зонирование повышает доступность приложений.

Если вместо зон доступности вы используете группы доступности, сначала создайте такую группу, а затем добавьте в нее виртуальные машины. После этого платформа Azure распределит ВМ по доменам сбоя и обновления. Чтобы максимально использовать высокую доступность Azure для своего магазина пиццы, выбирайте зоны доступности.

Попробуйте сейчас

Чтобы создать виртуальные машины и подключить их к подсистеме балансировки нагрузки, выполните приведенные ниже действия:

1 Создайте первую виртуальную машину и назначьте ее зоне доступности с помощью параметра `--zone 1`:

```
az vm create \
  --resource-group azuremolchapter8 \
  --name webvm1 \
  --image ubuntults \
  --size Standard_B1ms \
  --admin-username azuremol \
  --generate-ssh-keys \
  --zone 1 \
  --nics webnic1
```

2 Создайте вторую виртуальную машину и назначьте ее зоне доступности 2. Подключите второй виртуальный сетевой адаптер, созданный ранее, с помощью параметра `--nics webnic2`:

```
az vm create \
  --resource-group azuremolchapter8 \
  --name webvm2 \
  --image ubuntults \
  --size Standard_B1ms \
  --admin-username azuremol \
  --generate-ssh-keys \
  --zone 2 \
  --nics webnic2
```

Чтобы увидеть подсистему балансировки нагрузки в действии, установите базовый веб-сервер (как в главе 2). Вы также можете попробовать правило NAT подсистемы балансировки нагрузки. Уже видите взаимосвязь и взаимодействие этих компонентов в Azure?

Попробуйте сейчас

В главе 5 мы обсудили агент SSH. Агент SSH позволяет передавать ключ SSH с одной виртуальной машины на другую. Только ВМ 1 имеет правило NAT подсистемы балансировки нагрузки, поэтому для подключения к ВМ 2 требуется агент. Чтобы установить веб-сервер на виртуальные машины, выполните приведенные ниже действия:

1 Запустите агент SSH и добавьте ключ SSH, чтобы подключиться к обеим виртуальным машинам:

```
eval $(ssh-agent) && ssh-add
```

2 Получите публичный IP-адрес, подключенный к пулу внешних IP-адресов подсистемы балансировки нагрузки. Это единственный способ маршрутизации трафика через ВМ:

```
az network public-ip show \
  --resource-group azuremolchapter8 \
  --name publicip \
  --query ipAddress \
  --output tsv
```

3 Теперь вы можете подключиться по протоколу SSH к ВМ 1. Укажите публичный IP-адрес подсистемы балансировки нагрузки (замените `<your-ip-address>` в следующей команде) и порт, который использовался правилом NAT подсистемы балансировки нагрузки, например 50001. Параметр `-A` предписывает передавать ключи с помощью агента SSH:

```
ssh -A azuremol@<your-ip-address> -p 50001
```

В главе 2 вы использовали `apt-get` для установки всего стека LAMP, включавшего в себя веб-сервер Apache. Веб-сервер Apache немного отличается от автономного, но мощного веб-сервера NGINX. На виртуальной машине Windows обычно устанавливается IIS. Выполните следующую команду, чтобы установить веб-сервер NGI NX:

```
sudo apt update && sudo apt install -y nginx
```

4 В репозитории образцов GitHub из предыдущих глав есть базовая веб-страница HTML и страница проверки для зонда работоспособности, выполняемого подсистемой балансировки нагрузки. Скопируйте эти образцы на виртуальную машину:

```
git clone https://github.com/fouldsy/azure-mol-samples-2nd-ed.git
```

5 Скопируйте образец HTML-страницы и зонда работоспособности в каталог веб-сервера:

```
sudo cp azure-mol-samples-2nd-ed/08/webvm1/* /var/www/html/
```

6 Теперь вам нужно подключиться ко второй ВМ, установить веб-сервер NGINX и пример кода. Помните агент SSH? Вы сможете подключиться по протоколу SSH от ВМ 1 к ВМ 2 по внутреннему частному IP-адресу:

```
ssh 10.0.1.5
```

7 Установите веб-сервер NGINX:

```
sudo apt update && sudo apt install -y nginx
```

8 Скопируйте образцы с GitHub на виртуальную машину:

```
git clone https://github.com/fouldsy/azure-mol-samples-2nd-ed.git
```

9 Скопируйте образец HTML-страницы и зонда работоспособности в каталог веб-сервера:

```
sudo cp azure-mol-samples-2nd-ed/08/webvm2/* /var/www/html/
```

Откройте веб-браузер и перейдите по публичному IP-адресу подсистемы балансировки нагрузки. Загрузится базовая веб-страница, которая покажет, что теперь ваш магазин пиццы имеет избыточные ВМ в зонах доступности вне подсистемы балансировки нагрузки (см. рис. 8.11). Возможно, придется принудительно обновить веб-страницу, чтобы убедиться, что ВМ 1 и ВМ 2 реагируют на распределение трафика между ними.

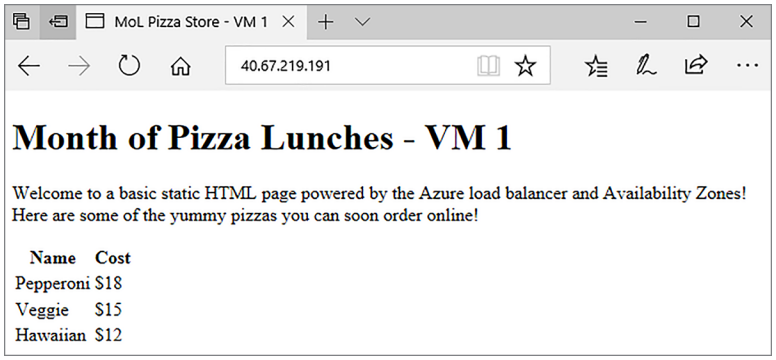


Рис. 8.11. Когда вы переходите в веб-браузере по публичному IP-адресу подсистемы балансировки нагрузки, трафик передается на одну из ВМ, на которой работает ваш основной веб-сайт. Зонд работоспособности, выполняемый подсистемой балансировки нагрузки, подтверждает ответы веб-сервера на странице health.html с помощью HTTP-кода 200 (OK). Если это так, виртуальная машина получает трафик, распределяемый подсистемой балансировки нагрузки.

8.3 *Практическое упражнение: рассмотрение шаблонов существующих развертываний*

Эта глава связывает воедино все, что вы узнали в предыдущих главах. Вы создали сетевые ресурсы, как в главе 5. С помощью зон обеспечили высокую доступность подсистемы балансировки нагрузки и виртуальных машин, как в главе 7. Вы установили веб-сервер и развернули образцы файлов, как в главе 2. Ваш магазин пиццы проделал долгий путь от базовой веб-страницы на одной ВМ в начале книги.

В этом практическом упражнении работе вы исследуете все ресурсы подсистемы балансировки нагрузки, чтобы опробовать материал предыдущей главы. Для этого вы рассмотрите шаблон Resource Manager из главы 6. Цель практического упражнения — показать, как один шаблон может создать и настроить то, на что потребовалось много страниц и команд CLI. И поверьте, команд PowerShell потребуется еще больше. Выполните приведенные ниже действия:

1. Откройте портал Azure.
2. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева.
3. Выберите группу ресурсов, например azuremolchapter8.
4. Выберите «Экспорт шаблона» на панели слева, как показано на рисунке 8.12.
5. Чтобы просмотреть соответствующую часть шаблона, выбирайте каждый ресурс в списке. Исследуйте, как представлены в шаблоне все ресурсы и компоненты, настроенные в Azure CLI.

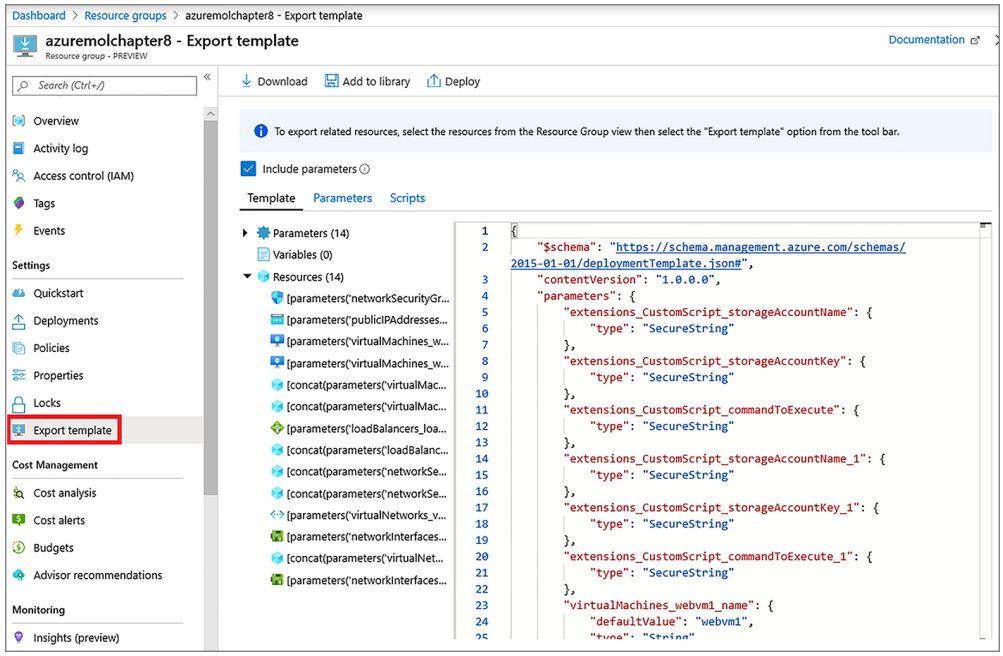


Рис. 8.12. На портале Azure выберите группу ресурсов подсистемы балансировки нагрузки и просмотрите шаблон Resource Manager.

Шаблон существенно упрощает развертывание избыточной и высокодоступной среды приложений с балансировкой нагрузки. Вы можете изменять имя, правила и режим распределения для подсистемы балансировки нагрузки, а также использовать шаблон, чтобы развернуть и настраивать всю среду приложений.

Не забудьте удалить эту группу ресурсов, чтобы максимально использовать бонусы от Azure.

9 Масштабируемые приложения

В 2 предыдущих главах мы рассмотрели создание приложений с высокой доступностью и использование подсистемы балансировки нагрузки для распределения трафика между несколькими виртуальными машинами, на которых выполняется ваше приложение. Но как эффективно управлять несколькими ВМ и запускать требуемое число экземпляров, когда клиенты нуждаются в них больше всего? Если спрос клиентов растет, вы хотите удовлетворить его, автоматически увеличивая масштаб приложения. Если же спрос падает (например, глубокой ночью, когда большинство людей спят), вы хотите уменьшить масштаб приложения и сэкономить деньги.

В Azure можно автоматически масштабировать ресурсы IaaS с помощью масштабируемых наборов виртуальных машин. Масштабируемые наборы запускают идентичные ВМ, обычно распределяемые вне подсистемы балансировки нагрузки или шлюза приложений. Вы определяете правила автоматического масштабирования, которые увеличивают или уменьшают число экземпляров ВМ при изменении спроса клиентов. Подсистема балансировки нагрузки или шлюз приложений автоматически распределяют трафик на новые экземпляры, что позволяет вам сосредоточиться на создании и выполнении приложений. Масштабируемые наборы позволяют управлять ресурсами IaaS благодаря гибкости PaaS. Веб-приложения, упомянутые в 2 предыдущих главах, теперь можно масштабировать в зависимости от спроса.

В этой главе мы рассмотрим проектирование и создание приложений с возможностью автоматического масштабирования. Вы узнаете, почему возможность масштабирования в зависимости от спроса обеспечивает эффективность приложений, и изучите способы масштабирования на основе различных показателей.

9.1 Зачем создавать масштабируемые и надежные приложения?

Что дает создание масштабируемых приложений? Вы удовлетворяете растущий спрос клиентов, даже когда смотрите кино на выходных. Вы не оплачиваете кучу дополнительных неиспользуемых ресурсов и приложение, которое не работает из-за их отсутствия. Оптимальный объем ресурсов для приложений редко бывает постоянным. Как правило, приложению требуются «приливы» и «отливы» днем и ночью, в будни и в выходные.

Существуют два основных способа масштабирования ресурсов: по вертикали и по горизонтали (см. рис. 9.1). Масштабируемые наборы ВМ и веб-приложения могут масштабироваться вертикально или горизонтально.

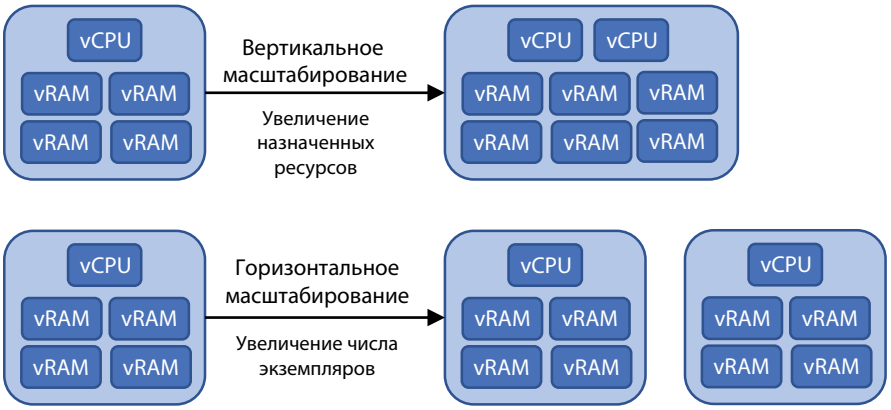


Рис. 9.1. Вы можете масштабировать приложения вверх и вниз или с уменьшением и с увеличением. Используемый способ зависит от обработки масштабирования в приложении. Вертикальное масштабирование корректирует ресурсы, назначенные виртуальной машине или веб-приложению, например число ядер ЦП или объем памяти. Этот способ масштабирования хорошо работает, если приложение выполняется только на одном экземпляре. Горизонтальное масштабирование изменяет число экземпляров, на которых выполняется приложение, повышая доступность и отказоустойчивость.

Масштабируемые приложения тесно связаны с приложениями высокой доступности. В главах 7 и 8 мы потратили много времени на группы и зоны доступности, а также на настройку подсистем балансировки нагрузки. Обе главы посвящены запуску нескольких виртуальных машин. При автоматическом масштабировании приложения его доступность тоже увеличивается, поскольку ВМ распределяются между группами или зонами доступности. Это здорово! Преимущество Azure в том, что вам не нужно беспокоиться о том, как добавить дополнительные экземпляры приложений и распределить их между оборудованием ЦОД (или даже между несколькими ЦОД). Кроме того, вам не нужно обновлять сетевые ресурсы для распределения трафика между новыми экземплярами приложения.

9.1.1 Масштабирование виртуальных машин по вертикали

Первый способ масштабирования ресурсов проверен временем, и вы, вероятно, использовали его раньше. Если приложение начинает работать медленно из-за увеличения числа клиентов, что вы обычно делаете? Добавляете ядра ЦП или память, не так ли? Для удовлетворения спроса вы масштабируете ресурс *вертикально*.

Чаще всего вертикальное масштабирование применяется для серверов баз данных. Базы данных, как известно, требовательны к вычислительным ресурсам — даже требовательнее, чем клиенты вашего магазина пиццы. Серверы баз данных часто потребляют все ресурсы виртуальной машины, даже если не используют их немедленно. Это может затруднить мониторинг фактических требований к системе, и вы не будете знать, когда нужно выполнить вертикальное масштабирование для предоставления дополнительных ресурсов. На рисунке 9.2 показан типичный ответ о вертикальном масштабировании для сервера баз данных, которому нужны дополнительные ресурсы.

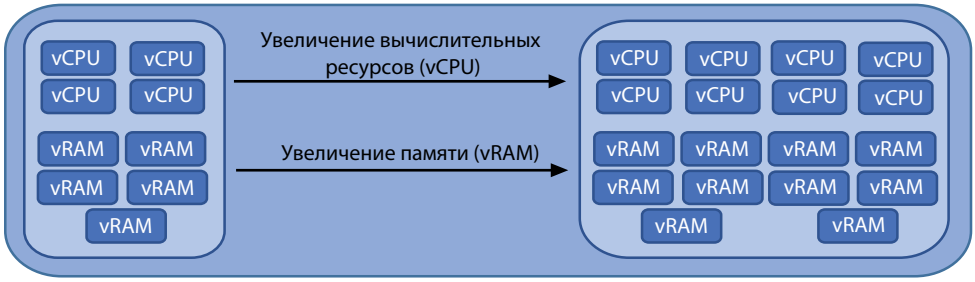


Рис. 9.2. По мере роста базе данных требуется все больше ресурсов для хранения и обработки данных в памяти. При таком сценарии вертикального масштабирования необходимо добавить ядра ЦП и память.

Вам может потребоваться масштабирование, превышающее спрос на процессор или память. Например, веб-сайт, который обрабатывает большой объем изображений или видео. В этом случае требований к обработке будет немного, зато ожидаются высокие требования к пропускной способности. Чтобы увеличить пропускную способность, вы должны увеличить число сетевых адаптеров на виртуальной машине. Если же нужно хранить больше изображений и видео, следует добавить внешнюю память. Эти ресурсы (виртуальные сетевые адаптеры и внешнюю память) можно добавлять или удалять в процессе работы ВМ.

Изменение размеров виртуальных машин

Azure позволяет увеличить размер ВМ (масштабировать вверх), если приложению требуются дополнительные вычислительные ресурсы. В главе 2 вы создали базовую виртуальную машину с размером, вероятно, Standard_D2s_v3. Это ничего не говорит вам о вычислительных ресурсах, назначенных ВМ, и о том, нужно ли добавлять ядра ЦП или память. Чтобы масштабировать по вертикали, требуется знать имеющиеся варианты.

Попробуйте сейчас

Чтобы увидеть доступные размеры виртуальных машин и вычислительные ресурсы, следуйте инструкциям, приведенным ниже:

1. Зайдите на портал Azure в браузере и откройте Cloud Shell.
2. Чтобы получить список доступных размеров ВМ и вычислительных ресурсов, которые они предоставляют, введите следующую команду Azure CLI:

```
az vm list-sizes --location eastus --output table
```

Выходные данные команды `az vm list-sizes` варьируются от региона к региону и меняются с течением времени, поскольку Azure корректирует семейства виртуальных машин. Этот короткий пример выходных данных показывает `MemoryInMb` (память в мегабайтах) и `NumberOfCores` (число ядер) для каждого размера ВМ:

```
MaxDataDiskCount    MemoryInMb    Имя                 NumberOfCores

               4          8192    Standard_D2s_v3                 2
               8         16384    Standard_D4s_v3                 4
              16         32768    Standard_D8s_v3                 8
              32         65536    Standard_D16s_v3               16
               8          4096    Standard_F2s_v2                 2
              16          8192    Standard_F4s_v2                 4
              32         16384    Standard_F8s_v2                 8
               2          2048    Standard_B1ms                   1
               2          1024    Standard_B1s                    1
               4          8192    Standard_B2ms                   2
               4          4096    Standard_B2s                    2
```

Итак, виртуальная машина Standard_D2s_v3 предоставляет 2 ядра ЦП и 8 ГБ памяти. Этого вполне достаточно для базовой ВМ, на которой работает веб-сервер. Если объем заказов в вашем магазине пиццы начинает расти и требуется вертикальное масштабирование. Вы можете использовать команду `az vm resize`, чтобы выбрать другой размер. Укажите размер виртуальной машины с необходимым числом ядер ЦП и объемом памяти.

Дополнительные ядра ЦП и память не появляются в виртуальной машине моментально. Этот процесс несколько отличается от использования Hyper-V или VMware в локальной среде. Ведь там можно добавлять или удалять основные вычислительные ресурсы в процессе работы ВМ Чтобы изменить размер виртуальной машины в Azure, требуется перезагрузить ее для регистрации новых вычислительных ресурсов и активации соответствующих правил выставления счетов. Для масштабирования по вертикали планируйте отключение на время перезагрузки ВМ.

Вертикальное уменьшение масштаба

Что делать, если объем ресурсов ВМ больше необходимого? Это более распространенный сценарий, чем виртуальная машина с нехваткой ресурсов. Зачастую владельцы приложений выбирают размер виртуальной машины с запасом, чтобы гарантировать бесперебойную работу своего приложения. Эти неиспользуемые ресурсы стоят денег, но остаются незамеченными, пока в конце месяца не приходит счет.

Ресурсы можно масштабировать в обоих направлениях. Мы рассмотрели масштабирование ресурсов *вверх*, но эти же концепции работают для масштабирования ресурсов *вниз*. Важно определить используемые размеры ВМ и спрос приложений на предоставленные ресурсы. Затем можно использовать команду `az vm resize` для выбора размера виртуальной машины с меньшим числом ядер ЦП и объемом памяти. Повторюсь, что на данный момент для любого изменения размера требуется перезагрузить ВМ.

9.1.2 Масштабирование веб-приложений по вертикали

Веб-приложения можно масштабировать вверх или вниз в зависимости от потребности в ресурсах аналогично виртуальным машинам. Создавая веб-приложение в главе 3, вы указали стандартный размер S1 Standard с одним ядром ЦП и 1,75 ГБ ОЗУ. Каждая ценовая категория (размер) предоставляет определенное количество ресурсов: ядер ЦП, памяти и промежуточных слотов. Даже если изменяются стандартные размер и выделение ресурсов либо выбирается другой размер, концепция не меняется.

Если вы создаете веб-приложение и считаете, что оно требует больше ресурсов, чем предусмотрено сервисным планом, выберите другую ценовую категорию (см. рис. 9.3). Поступайте так же при избытке ресурсов. Таким образом, при необходимости вы можете вручную масштабировать свое веб-приложение вверх или вниз.

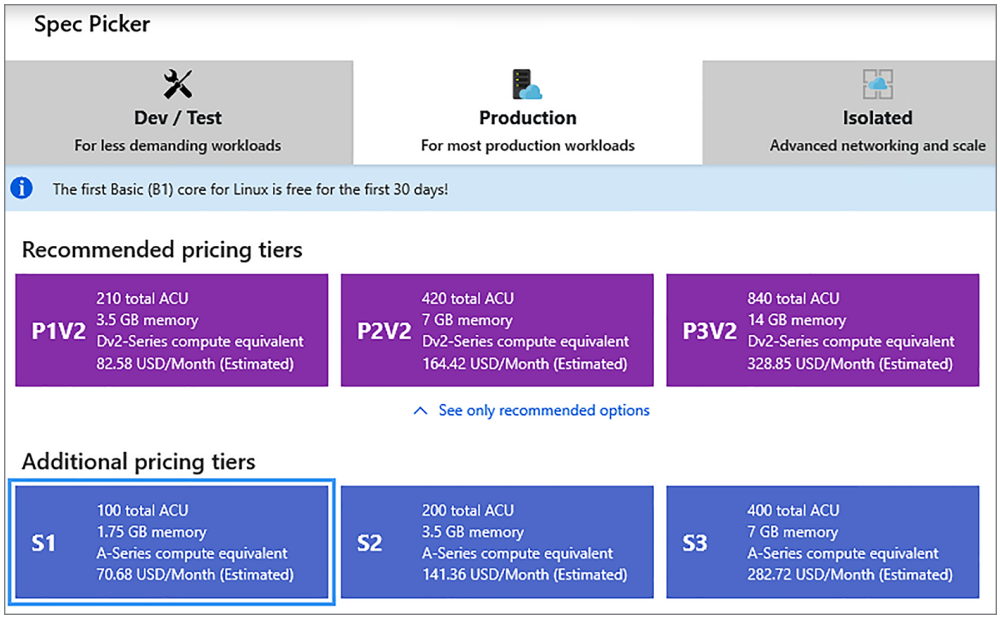


Рис. 9.3. Для масштабирования веб-приложения по вертикали вручную измените ценовую категорию (размер) сервисного плана. Сервисный план определяет количество ресурсов, назначенных веб-приложению. Если приложению требуются другой объем внешней памяти, другое число ядер ЦП или слотов развертывания, можно сменить категорию, чтобы назначить соответствующие ресурсы.

9.1.3 Масштабирование ресурсов по горизонтали

Другой способ удовлетворить растущий спрос — масштабирование по горизонтали с увеличением. Для масштабирования по вертикали вы увеличиваете число ядер ЦП и объем памяти для одного ресурса, например для ВМ. Для масштабирования по горизонтали вы увеличиваете число виртуальных машин (см. рис. 9.4).

Горизонтально масштабируемое приложение должно поддерживать такую функцию и обрабатывать данные без конфликтов. Веб-приложения, как правило, могут сами обрабатывать данные, поэтому они отлично подходят для масштабирования по горизонтали.

При создании сложных приложений можно разбивать их на отдельные компоненты. Применительно к очередям хранилища Azure из главы 4, можно

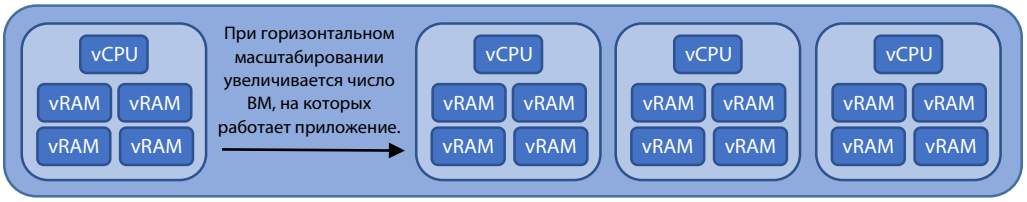


Рис. 9.4. Чтобы удовлетворить растущий спрос, можно увеличить число виртуальных машин, на которых выполняется приложение, при этом нагрузка распределяется между несколькими виртуальными машинами, а не на крупный экземпляр ВМ.

иметь один компонент приложения, который получает веб-заказы, и другой компонент, который обрабатывает и передает их в магазин пиццы. Использование очередей сообщений — это один из подходов к проектированию и созданию приложений, которые могут работать в горизонтально масштабируемой среде. Помимо прочего, этот подход позволяет масштабировать каждый компонент приложения отдельно и использовать различные размеры ВМ или планы веб-приложений для повышения эффективности и уменьшения ежемесячного счета.

Раньше вы предпочитали масштабировать по вертикали, поскольку было проще добавить вычислительные ресурсы в надежде, что их будет достаточно для приложения. Физическая настройка кластера ресурсов и масштабирование приложения по горизонтали зачастую были очень сложными. Облачные вычисления и виртуализация настолько упростили горизонтальное масштабирование, что оно часто быстрее вертикального и выполняется без простоев.

Помните команду `az vm resize` из этой главы? Что происходит после изменения размера ВМ? Она перезапускается. Если выполняется единственный экземпляр вашего приложения, никто не сможет обратиться к нему, пока виртуальная машина не возобновит работу. А при горизонтальном масштабировании экземпляры ВМ добавляются без простоев — новые виртуальные машины сразу начинают обрабатывать запросы приложения. Зонды работоспособности в подсистеме балансировки нагрузки, рассмотренные в главе 8, автоматически определяют, когда новая ВМ во внутреннем пуле готова обрабатывать запросы клиентов, и трафик начинает поступать на нее.

Azure обеспечивает гибкость и возможность выбирать способ масштабирования. Если вы разрабатываете новую среду приложений, я рекомендую использовать горизонтальное масштабирование. В этом помогут близкие родственники виртуальных машин Azure: масштабируемые наборы виртуальных машин.

9.2 Масштабируемые наборы виртуальных машин

Виртуальные машины небезосновательно считаются наиболее распространенными рабочими нагрузками в Azure. Научиться создавать и запускать виртуальную машину несложно, поскольку большая часть того, что вы уже знаете, подходит и для Azure. Веб-серверы — это наиболее распространенные рабочие нагрузки для ВМ, что тоже понятно, поскольку запуск Apache, IIS или NGINX на виртуальной машине Azure не требует новых навыков.

Что если веб-сервер запускается в кластере виртуальных машин? Сможете проделать это в обычной локальной среде? Ведь выбор кластерных решений огромен. А что насчет обновлений физических серверов или ВМ? Как вы справитесь с ними? А если вы хотите автоматически увеличивать или уменьшать число экземпляров в кластере? Может быть, вам нужен другой инструмент? На рисунке 9.5 показана схема масштабируемого набора виртуальных машин.

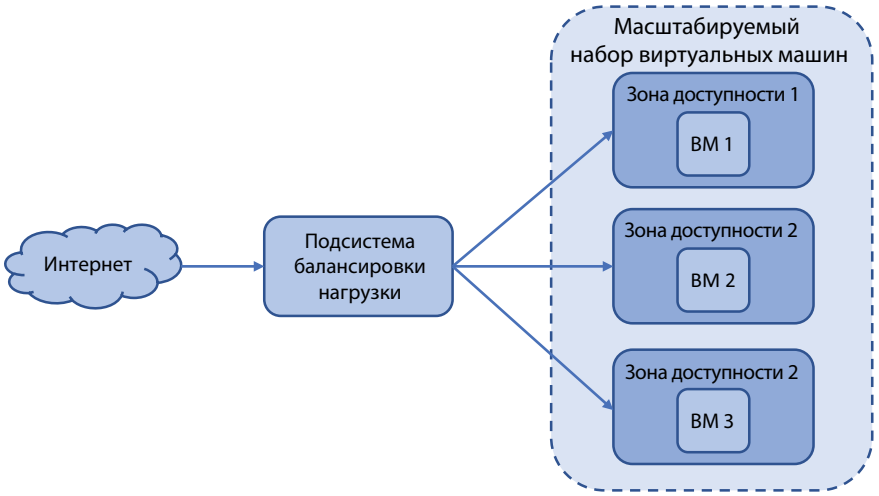


Рис. 9.5. Масштабируемый набор логически группирует виртуальные машины. Виртуальные машины идентичны и могут централизованно управляться, обновляться и масштабироваться. Вы можете определить показатели для автоматического увеличения или уменьшения числа ВМ в масштабируемом наборе в зависимости от нагрузки приложения.

Масштабируемый набор упрощает управление несколькими ВМ, обеспечивая высокую доступность и балансировку нагрузки приложения. Вы указываете Azure размер виртуальной машины, ее базовый образ и число экземпляров. Затем можно определить показатели ЦП или памяти для автоматического увеличения или уменьшения числа экземпляров в зависимости от нагрузки приложения или в часы пик согласно расписанию. Масштабируемые наборы дополняют модель IaaS виртуальных машин возможностями PaaS, такими как масштабирование, избыточность, автоматизация и централизованное управление ресурсами.

Масштабируемый набор с одной виртуальной машиной?

Создавая приложения на виртуальных машинах, используйте масштабируемый набор, даже если вам нужна всего одна ВМ. Почему? Масштабируемый набор может расширяться в любое время и автоматически создает подключения к подсистеме балансировки нагрузки или шлюзу приложений. Если спрос на приложение внезапно увеличивается на протяжении двух месяцев, можно указать, чтобы масштабируемый набор создал дополнительный экземпляр виртуальной машины (или 2).

Для развертывания обычной автономной ВМ необходимо добавить ее в подсистему балансировки нагрузки, а если ВМ не входит в группу или зону доступности, вы должны обеспечить высокую доступность этой машины. Создавая масштабируемый набор с нуля (даже для одной ВМ), вы с минимальными затратами подготавливаете свое приложение к будущим изменениям.

9.2.1 Создание масштабируемого набора виртуальных машин

Хотя масштабируемый набор упрощает создание и выполнение приложений с высокой доступностью, необходимо создать и настроить несколько новых компонентов. Однако с помощью Azure CLI масштабируемый набор развертывается всего двумя командами.

Попробуйте сейчас

Чтобы создать масштабируемый набор с помощью Azure CLI, выполните приведенные ниже действия:

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Создайте группу ресурсов с помощью команды `az group create`. Укажите имя группы ресурсов, например `azuremolchapter11`, и расположение:

   ```
   az group create --name azuremolchapter9 --location westeurope
   ```

   Масштабируемые наборы могут использовать зоны доступности, поэтому выберите регион, поддерживающий эту функцию.
3. Чтобы создать масштабируемый набор, укажите число экземпляров ВМ и то, как они обрабатывают обновления своей конфигурации. При изменении ВМ, например установке приложения или обновлении гостевой ОС, виртуальные машины автоматически обновляются после обнаружения изменений. Вы также можете установить политику обновлений manual и применять обновления в удобное для вас время. Остальные параметры знакомы вам по созданию одной виртуальной машины:

   ```
   az vmss create \
     --resource-group azuremolchapter9 \
     --name scalesetmol \
     --image UbuntuLTS \
     --admin-username azuremol \
     --generate-ssh-keys \
     --instance-count 2 \
     --vm-sku Standard_B1ms \
     --upgrade-policy-mode automatic \
     --lb-sku standard \
     --zones 1 2 3
   ```

Вот и все! Вы создали несколько виртуальных машин в масштабируемой зоне доступности. Но есть кое-что действительно потрясающее в масштабируемом наборе, который вы только что получили с помощью Azure CLI. Помните, в главе 8 мы рассмотрели подсистемы балансировки нагрузки и используемые команды CLI, а также шаблоны, упрощающие создание такой подсистемы? Это команда `az vmss create` создала и настроила подсистему балансировки нагрузки для вас!

Не забывайте о квотах

Я уже говорил об этом в главе 7, но во избежание проблем стоит повторить. В Azure стандартные квоты на подписку не позволят случайно развернуть ресурсы и забывать о них — это будет стоить слишком дорого. Список квот приведен на странице http://mng.bz/ddcx.

При создании нескольких ВМ могут возникнуть проблемы с квотами. Кроме того, не забывайте удалять ресурсы из предыдущих глав и упражнений. Если вы видите текст

```
Операция приводит к превышению лимитов квоты ядер .
Максимально допустимое количество: 4, используется сейчас: 4, запрошено
дополнительно: 2.
```

значит нужно запросить дополнительны квоты. Чтобы просмотреть текущую квоту для данного региона, выполните следующую команду:

```
az vm list-usage --location westeurope
```

Чтобы запросить дополнительные квоты для региона, выполните действия, описанные на странице http://mng.bz/Xq2f.

Вы создали масштабируемый набор, используя минимальное число запросов в Azure CLI. Кроме того, была создана и настроена подсистема балансировки нагрузки, назначен публичный IP-адрес, а экземпляры ВМ масштабируемого набора добавлены в пул внутренних IP-адресов.

Попробуйте сейчас

Ознакомьтесь с ресурсами, созданными с помощью масштабируемого набора, как описано далее.

Чтобы узнать, какие ресурсы были созданы с помощью масштабируемого набора, выполните следующую команду:

```
az resource list \
  --resource-group azuremolchapter9 \
  --output table
```

Пример выходных данных приведен ниже. Данные в столбце Type показывают, что созданы виртуальная сеть, публичный IP-адрес и подсистема балансировки нагрузки:

```
Имя       ResourceGroup     Тип
--------  ----------------  ----------------------------------------
mol       azuremolchapter9  Microsoft.Compute/virtualMachineScaleSets
molLB     azuremolchapter9  Microsoft.Network/loadBalancers
molLBIP   azuremolchapter9  Microsoft.Network/publicIPAddresses
molVNET   azuremolchapter9  Microsoft.Network/virtualNetworks
```

Что означает все это волшебство? Создавая масштабируемый набор с помощью Azure CLI, вы получаете избыточную в пределах зоны подсистему балансировки нагрузки и общедоступный IP-адрес. Кроме того, создаются виртуальные машины, которые добавляются в пул внутренних IP-адресов в подсистеме балансировки нагрузки. Создаются правила NAT, позволяющие подключаться к экземплярам ВМ. Не хватает только правил подсистемы балансировки нагрузки, поскольку они зависят от

выполняемых приложений. Если вы добавляете ВМ в масштабируемый набор или удаляете ее оттуда, конфигурация балансировки нагрузки автоматически обновляется и разрешает распределять трафик на новые экземпляры. Но волшебство присуще не только Azure CLI — если вы используете Azure PowerShell или портал Azure, создаются и подключаются вспомогательные сетевые ресурсы.

Попробуйте сейчас

Созданный вами масштабируемый набор состоит из двух экземпляров. Вы можете вручную изменить число экземпляров ВМ в масштабируемом наборе. При этом подсистема балансировки нагрузки автоматически обновит конфигурацию пула внутренних IP-адресов. Укажите четыре экземпляра в параметре `--new-capacity` масштабируемого набора:

```
az vmss scale \
    --resource-group azuremolchapter9 \
    --name scalesetmol \
    --new-capacity 4
```

9.2.2 Создание правил автоматического масштабирования

При создании масштабируемого набора вы развернули фиксированное число экземпляров. Одна из важнейших функций наборов масштабирования — автоматическое увеличение или уменьшение числа экземпляров ВМ, запущенных в наборе масштабирования.

Как показано на рисунке 9.6, число экземпляров в масштабируемом наборе автоматически увеличивается с ростом нагрузки приложения. Представьте себе типичное бизнес-приложение в своей среде. В начале рабочего дня пользователи начинают обращаться к приложению, что приводит к росту загрузки ресурсов на экземпляры виртуальных машин. Чтобы обеспечить оптимальную производительность приложения, масштабируемый набор автоматически добавляет экземпляры виртуальных машин. Подсистема балансировки нагрузки начинает автоматически распределять трафик на новые экземпляры. В конце рабочего дня, когда пользователи уходят домой, спрос на приложение падает. ВМ потребляют меньше ресурсов, поэтому масштабируемый набор автоматически удаляет некоторые экземпляры, чтобы уменьшить объем ненужных ресурсов и снизить расходы.

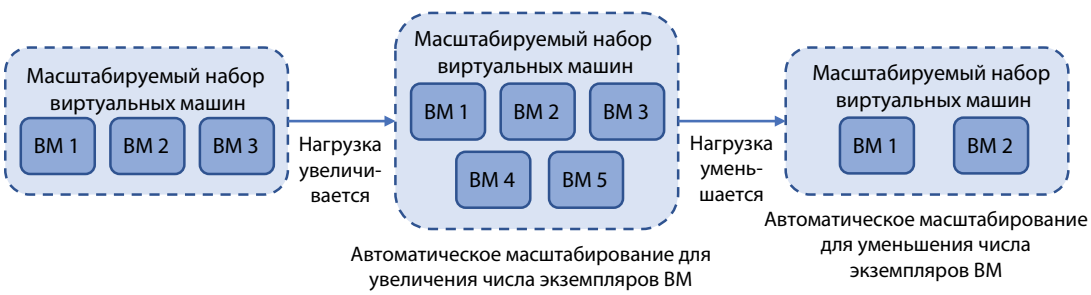


Рис. 9.6. Масштабируемые наборы могут автоматически уменьшаться и увеличиваться. Вы определяете правила мониторинга определенных показателей, которые активируют правила увеличения или уменьшения числа выполняемых экземпляров ВМ. По мере изменения спроса на приложение изменяется число экземпляров ВМ. Это позволяет максимально увеличить производительность и доступность приложения, а также свести к минимуму ненужные расходы при снижении нагрузки приложения.

В правилах масштабируемых наборов можно использовать различные показатели. Вы можете анализировать показатели хоста для потребления базовых ресурсов, настроить коллекцию показателей гостевой ВМ, чтобы анализировать конкретные счетчики производительности приложения, или запустить Azure Application Insights для глубокого мониторинга кода приложения.

Кроме того, вы можете использовать расписания, чтобы определять число экземпляров ВМ в масштабируемом наборе во временном окне. В примере обычного бизнес-приложения, спрос на которое в дневное время выше, чем в вечернее, можно увеличить фиксированное число экземпляров, выполняемых в дневное время, и определить фиксированное число экземпляров, выполняемых в вечернее время.

Правила автоматического масштабирования на основе показателей отслеживают производительность за определенный интервал (например, за 5 минут), а затем при необходимости в течение нескольких минут создают и настраивают новые экземпляры ВМ. Если вы автоматически масштабируете число экземпляров ВМ в масштабируемом наборе с помощью постоянных расписаний, дополнительные машины создаются заранее и подсистема балансировки нагрузки распределяет трафик на них в течение дня.

При использовании расписаний учитывается базовый показатель обычного спроса на приложение и не учитывается повышенный или пониженный спрос на определенных участках учетной записи или цикла продаж. В конечном итоге вы можете получить избыток ресурсов и переплатить за них. Кроме того, возможна ситуация, когда нагрузка приложения на экземпляры ВМ в масштабируемом наборе превышает допустимую.

Попробуйте сейчас

Чтобы создать правила автоматического масштабирования для масштабируемого набора, выполните приведенные ниже действия:

1. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2. Выберите группу ресурсов, созданную для развертывания шаблона, например `azuremolchapter9`.
3. Выберите масштабируемый набор в списке ресурсов, например scalesetmol.
4. В разделе «Параметры» слева в окне «Масштабируемый набор» нажмите «Масштабирование». Вы можете масштабировать или создавать собственные пользовательские правила автомасштабирования вручную.
5. Выберите для создания пользовательских правил автомасштабирования.
6. Введите имя, например `autoscale`, а затем определите минимальное, максимальное и стандартное число экземпляров. Для этого упражнения: минимальное — 2, максимальное — 10, стандартное — 2.
7. Нажмите «Добавить правило» и просмотрите доступные параметры правила (см. рис. 9.7).

   Стандартные параметры отрегулированы согласно средней загрузке ЦП. Правило срабатывает, если нагрузка превышает 70 % в течение 10-минутного интервала. Масштабируемый набор увеличивается на 1 экземпляр ВМ, а правила остаются в состоянии ожидания 5 минут, после чего запускается мониторинг и может сработать следующее правило.

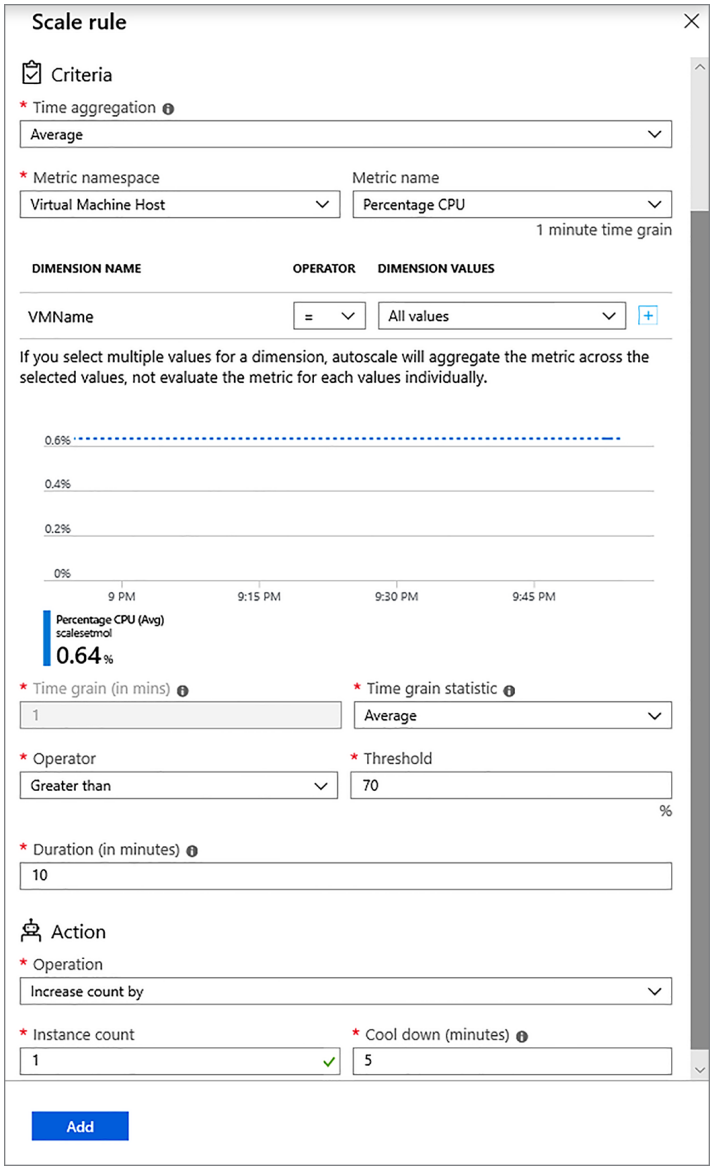


Рис. 9.7. При добавлении правила автоматического масштабирования следует указывать точные условия его срабатывания.

Это период «остывания», в течение которого новый экземпляр ВМ развертывается и начинает получать трафик от подсистемы балансировки нагрузки, что снижает общую нагрузку приложения на масштабируемый набор. Без периода остывания правила могут добавить еще один экземпляр виртуальной машины, прежде чем начнется распределение нагрузки на созданный ранее экземпляр.

8 Чтобы создать правило, нажмите кнопку «Добавить».

9 Выберите добавление другого правила. Добавьте правило «Уменьшить число на 1» для средней загрузки ЦП менее 30 % в течение 5 минут.

10 Проверьте правила (см. рис. 9.8) и нажмите кнопку «Сохранить».

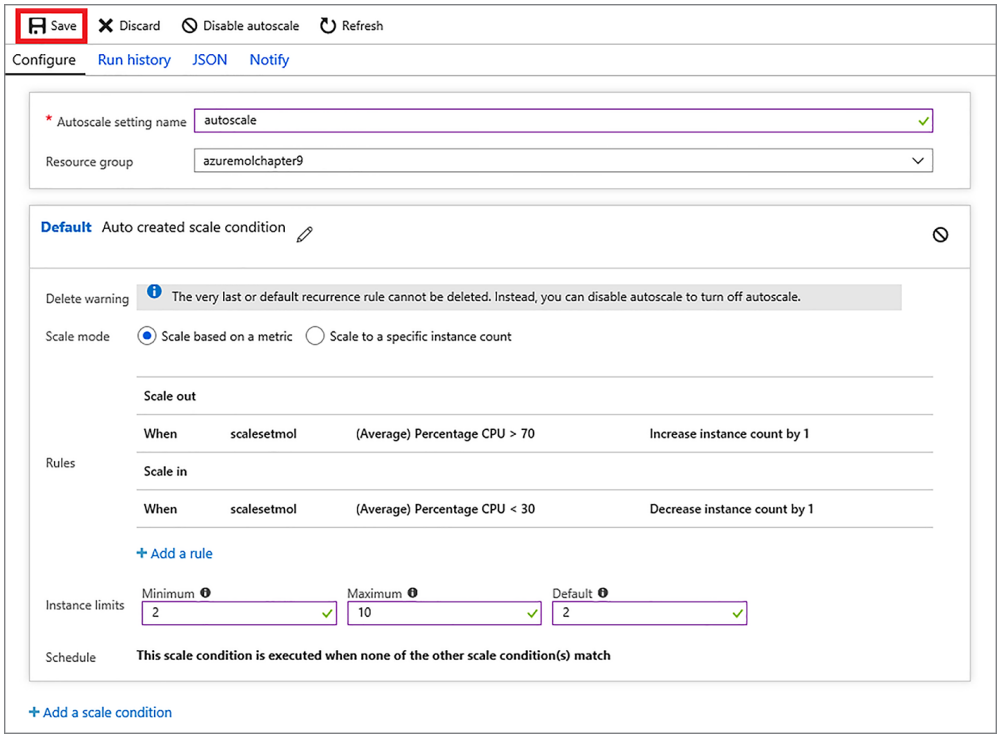


Рис. 9.8. Теперь у вас есть два правила. Первое увеличивает число экземпляров на 1 при средней загрузке ЦП выше 70 %, а второе уменьшает число экземпляров на 1, если средняя загрузка ЦП ниже 30 %.

Правила автоматического масштабирования также можно настроить с помощью Azure CLI, Azure PowerShell или в шаблонах. Портал — это отличный способ просматривать правила и варианты для каждого параметра. Создавая сложные правила, можно использовать шаблоны, которые позволяют создавать масштабируемые наборы с одним и тем же набором правил. Это обусловлено механизмом повторения.

9.3 Масштабирование веб-приложения

Если вы с интересом изучали веб-приложения в главе 3 или таблицы и очереди Azure в главе 4, то три последние, очень сложные главы о виртуальных машинах IaaS могли вызвать недоумение. Разве облако не должно быть проще? Для таких компонентов PaaS, как веб-приложения, — однозначно!

Однако не следует думать, что на следующих страницах рассказывается, как обеспечить такие же возможности высокой доступности и автоматического масштабирования для веб-приложений. На самом деле, все намного проще! Как обычно, выбор между IaaS и PaaS — это баланс между гибкостью и простотой управления. Значительная часть базовой избыточности абстрагируется в сервисах PaaS, например в веб-приложениях, поэтому главы о высокой доступности и подсистемах балансировки нагрузки не нужны.

Создание и выполнение ВМ или масштабируемых наборов с использованием подсистем балансировки нагрузки и зон доступности с помощью модели IaaS зависит от потребностей или ограничений бизнеса. Разработчики, инженеры по эксплуатации, инструменты и рабочие процессы могут быть не готовы к переходу на веб-приложения. Тем не менее, я настоятельно рекомендую подумать об использовании веб-приложений для новых развертываний. Компоненты PaaS, например веб-приложения, позволяют сосредоточиться на приложениях и клиентах, а не на инфраструктуре и администрировании.

Попробуйте сейчас

Чтобы создать веб-приложение с помощью Azure CLI, выполните приведенные ниже действия:

1. В главе 3 вы создали веб-приложение с помощью портала Azure. Как и для большинства ресурсов, это быстрее и проще реализуется с помощью Azure CLI. Откройте Cloud Shell на портале Azure.
2. Создайте план App Service (размер — S1 Standard). Этот размер позволяет автоматически масштабировать число экземпляров веб-приложения до 10:

```
az appservice plan create \
  --name appservicemol \
  --resource-group azuremolchapter9 \
  --sku s1
```

3. Создайте веб-приложение, использующее для развертывания локальный репозиторий Git (см. главу 3):

```
az webapp create \
  --name webappmol \
  --resource-group azuremolchapter9 \
  --plan appservicemol \
  --deployment-local-git
```

Все концепции и сценарии для правил автоматического масштабирования и расписаний для масштабируемых наборов, рассмотренные в разделе 9.2.2, применимы и к веб-приложениям. Вспомним несколько распространенных сценариев автоматического масштабирования веб-приложений:

- Автоматическое увеличение или уменьшение числа экземпляров веб-приложения на основе показателей производительности для удовлетворения спроса на приложение в течение рабочего дня.
- Расписание для автоматического увеличения числа экземпляров веб-приложения в начале рабочего дня и уменьшения в его конце.

В случае магазина пиццы веб-приложение может получать больше трафика в конце рабочего дня и вечером, поэтому не существует идеального набора правил автоматического масштабирования для любой ситуации. Опять же, необходим базовый показатель производительности приложения, чтобы понять, как оно работает в обычном режиме, и показатель производительности, при котором должно выполняться горизонтальное масштабирование приложения. Даже если вы используете расписания автоматического масштабирования, необходимо продолжать мониторинг пикового спроса на приложение, чтобы создать правила для этого варианта использования.

Попробуйте сейчас

Чтобы создать правила автоматического масштабирования для веб-приложения, выполните приведенные ниже действия:

1. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2. Выберите группу ресурсов, созданную для веб-приложения, например azuremolchapter9.
3. Выберите веб-приложение в списке ресурсов, например webappmol.
4. В разделе «Параметры» слева в окне веб-приложения выберите пункт «Расширить (план службы приложений)».
5. Снова выберите настройку пользовательских правил автомасштабирования, а не просто вручную масштабируйте веб-приложение.
6. Введите имя, например `autoscalewebapp`, а затем определите минимальное, максимальное и стандартное число экземпляров. Для этого упражнения: минимальное — 2, максимальное — 5, стандартное — 2.
7. Нажмите кнопку «Добавить правило» и просмотрите доступные параметры правила. Это окно выглядит так же, как окно правил автоматического масштабирования для масштабируемых наборов. Стандартные параметры настроены на среднюю загрузку ЦП, а правило срабатывает, если нагрузка превышает 70 % в течение 10-минутного интервала. Число экземпляров веб-приложения увеличивается на 1, а правила остаются в состоянии ожидания 5 минут, после чего начинается мониторинг и может сработать следующее правило.
8. Выберите добавление другого правила. Добавьте правило «Уменьшить число на 1» для средней загрузки ЦП меньше 30 % в течение 5 минут.
9. Просмотрите и сохраните правила.

Когда правила автоматического масштабирования активируют масштабирование веб-приложения с увеличением или с уменьшением, платформа Azure распределяет трафик по доступным экземплярам веб-приложения. Вам недоступна подсистема балансировки нагрузки, как в случае с масштабируемыми наборами, но трафик по-прежнему автоматически распределяется между экземплярами веб-приложений при горизонтальном масштабировании среды. Эта концепция аналогична предыдущей, но здесь все абстрагировано потому что при применении подхода PaaS вам не нужно ни о чем беспокоиться!

Как и масштабируемые наборы, веб-приложения позволяют создавать правила, автоматически масштабирующие число экземпляров вашего приложения. Кроме того, наличие нескольких экземпляров увеличивает доступность приложения. Масштабируемые наборы — это хороший компромисс между разработчиками и лицами, принимающими решения, которые хотят или вынуждены создавать приложения на ВМ с использованием PaaS-функций для автоматического масштабирования и перенаправления трафика клиентов.

В главе 11 мы рассмотрим диспетчер трафика Azure, который выполняет эти развертывания с высоким уровнем доступности. Сейчас же вы не совсем готовы к производственной среде с точки зрения нескольких избыточных масштабируемых наборов или экземпляров веб-приложений с автоматически распределенным трафиком между ними. Мы скоро обратимся к этому вопросу.

9.4 Практическое упражнение: установка приложений в масштабируемом наборе или веб-приложении

Мы многое рассмотрели в этой главе, и теперь вы можете выбрать финальное практическое упражнение: по масштабируемым наборам либо по веб-приложениям. Если же вы хотите продлить свой обеденный перерыв, сделайте обе!

9.4.1 Масштабируемые наборы виртуальных машин

У вас есть несколько экземпляров ВМ в масштабируемых наборах, но сейчас они малополезны. Обзор различных способов установки приложений на экземпляры ВМ в масштабируемом наборе приведен на странице http://mng.bz/9Ocx. На практике обычно используется один из методов автоматизированного развертывания, но сейчас вручную установите веб-сервер на экземплярах ВМ (как в главе 8):

1. Помните правила NAT подсистемы балансировки нагрузки? По умолчанию каждый экземпляр ВМ в масштабируемом наборе имеет правило NAT, позволяющее подключаться к нему напрямую по SSH. Порты не являются стандартным TCP-портом 22. Чтобы просмотреть список экземпляров ВМ в масштабируемом наборе и номера их портов, выполните следующую команду:

```
az vmss list-instance-connection-info \
  --resource-group azuremolchapter9 \
  --name scalesetmol
```

2. Для подключения к определенному порту по SSH используйте параметр `-p` (укажите свой публичный IP-адрес и номер порта):

```
ssh azuremol@40.114.3.147 -p 50003
```

3. Установите базовый веб-сервер NGINX на каждом экземпляре виртуальной машины с помощью команды `apt install`. Вспомните, как вы делали это в главе 8.
4. Чтобы увидеть набор масштабирования в действии, перейдите в веб-браузере по публичному IP-адресу для подсистемы балансировки нагрузки масштабируемого набора.
5. Если возникнут проблемы, убедитесь, что подсистема балансировки нагрузки правильно создала правило для порта 80 и что существует связанный зонд работоспособности для TCP-порта 80или собственный зонд для /health.html на ВМ.

9.4.2 Веб-приложения

Процесс развертывания приложения в несколько экземпляров веб-приложения аналогичен процессу для одного веб-приложения (см. главу 3). Вы отправляете приложение в локальный репозиторий Git, а платформа Azure, благодаря возможностям PaaS, развертывает единую базу кода в несколько экземпляров веб-приложения:

1. Инициализируйте репозиторий Git в azure-mol-samples-2nd-ed/09, а затем добавьте и зафиксируйте образцы файлов, как в главе 3:

```
cd azure-mol-samples-2nd-ed/09
git init && git add . && git commit -m "Pizza"
```

2. Вашему веб-приложению выделен локальный репозиторий Git. Добавьте удаленный репозиторий для своего веб-приложения (см. главу 3):

```
git remote add webappmolscale <your-git-clone-url>
```

3. Отправьте образец в свое веб-приложение. При этом происходит фиксация кода, а затем ваше приложение распределяется между несколькими экземплярами веб-приложения:

```
git push webappmolscale master
```

10 Глобальные базы данных с Cosmos DB

Данные. Без них не обойтись. Практически все приложения, которые вы создаете и запускаете, создают, обрабатывают или извлекают данные. Традиционно все эти данные хранились в структурированных базах данных, таких как MySQL, Microsoft SQL или PostgreSQL. Это большие структурированные базы данных, которые давно используются и всем известны; по ним имеется множество документации и учебников, а доступ к ним можно осуществлять на большинстве основных языков программирования.

Большие возможности — это большая ответственность, и, как правило, эксплуатация таких традиционных структурированных баз данных сопряжена со значительными инфраструктурными издержками и высокими требованиями к управлению. Мы отнюдь не хотим сказать, что вы не должны ими пользоваться — напротив! Однако когда речь идет о выполняемых в глобальном масштабе приложениях, создать кластеры серверов баз данных, которые бы реплицировали ваши данные и интеллектуально перенаправляли клиентов в ближайший экземпляр вашего решения, очень не легко.

И в этих сценариях Azure Cosmos DB может стать вашим лучшим другом. Вам не нужно переживать о реплицировании данных, обеспечении согласованности и распределении клиентских запросов. Вы добавляете данные с помощью любой из многочисленных доступных моделей, а затем выбираете, где должны быть доступны ваши данные. В этой главе мы расскажем вам о моделях неструктурированных баз данных в Cosmos DB, создании и настройке баз данных для глобального распределения, а также о разработке веб-приложений, использующих экземпляры Cosmos DB, которые отличаются высокой степенью резервирования и масштабируемости.

10.1 Что такое Cosmos DB?

В главе 4 мы начали изучать неструктурированные базы данных с таблицами хранилищ Azure. Несмотря на то что тот пример достаточно прост, рассмотренные нами концепции лежат в основе Cosmos DB. Во-первых, давайте четко сформулируем определения *структурированных* и *неструктурированных* баз данных.

10.1.1 Структурированные базы данных (SQL)

Структурированные базы данных — это наиболее традиционный подход к хранению данных. *Структура* или *схема* базы данных определяет способ представления данных. Данные хранятся в таблицах, и каждая строка представляет один элемент и фиксированный набор присвоенных ему значений. Если взять для примера модель магазина пиццы, то каждая строка в таблице с указанием типов пиццы может содержать название, размер и стоимость пиццы. Базовая база данных SQL показана на рисунке 10.1.

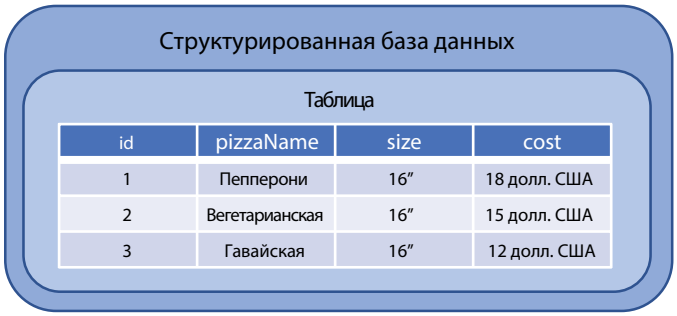


Рис. 10.1. В структурированной базе данных данные хранятся в строках и столбцах таблицы. Каждая строка содержит фиксированный набор столбцов, которые представляют собой схему этой базы данных.

В структурированных базах данных, как правило, на каждом сервере должна храниться вся база данных — только в этом случае запросы и операции извлечения данных будут успешными. Данные объединяются в запросы для извлечения из разных таблиц на основе критериев, созданных разработчиком в составе структурированного запроса. Поэтому такие базы данных называют базами данных *SQL — Structured Query Language (язык структурированных запросов)*. По мере увеличения масштабов и сложности баз данных серверы, на которых выполняются эти базы данных, должны масштабироваться и приобретать соответствующие ресурсы для содержания этих данных в памяти. Если наборы данных становятся слишком большими, поддерживать ресурсы на нужном уровне становится сложно и дорого. Учитывая необходимость сохранять структуру базы данных, в дальнейшем усложняется добавление свойств и изменение структуры базы данных.

10.1.2 Неструктурированные базы данных (NoSQL)

Неструктурированные данные в базах данных NoSQL хранятся не в таблицах, состоящих из строк и столбцов, а в динамических массивах, которые обеспечивают возможность добавления при необходимости в элемент новых свойств. Одним из значительных преимуществ такого подхода является возможность быстро добавить новый тип пиццы или особую начинку, не меняя базовую структуру базы данных. В структурированной базе данных потребовалось бы добавить в таблицу новый столбец, а затем обновить приложение, чтобы оно могло обрабатывать дополнительный столбец. В базах данных NoSQL новое свойство добавляется к записи из кода; см. рис. 10.2.

Рис. 10.2. В неструктурированной базе данных данные хранятся без фиксированного сопоставления столбцов строке в таблице. Чтобы добавить новую начинку к пицце, например, не нужно обновлять всю схему и другие записи.

Базы данных NoSQL также предоставляют разные модели баз данных. Эти модели дают представление о том, как хранятся и извлекаются данные в базе данных. Выбор модели зависит от размера и формата данных, с которыми вы работаете, а также от того, как нужно представить данные в своем приложении. Речь идет о следующих моделях: документ, график и таблица. Пока не нужно углубляться в эти модели слишком сильно; разным наборам неструктурированных данных больше подходят разные модели — это зависит от того, какие связи между данными вам нужно установить и какие запросы требуется адресовать этим данным. Основной вывод заключается в том, что в неструктурированных базах данных NoSQL реализован другой подход к хранению и извлечению данных, который можно использовать с выгодой для себя при разработке и запуске облачных приложений в Azure.

10.1.3 Масштабирование баз данных

Помните, я говорил, что если речь идет о структурированной базе данных, то, как правило, полная база данных должна быть размещена на каждом сервере? По мере того как базы данных растут, для их обслуживания требуются серверы большего размера. Возможно, вам никогда не доведется работать с базами данных, увеличивающимися до сотен ГБ или даже ТБ, однако нельзя не отметить, в базах данных NoSQL подход к росту и масштабированию баз данных реализован иначе, чем в базах данных SQL. Разница в том, что базы данных NoSQL, как правило, масштабируются горизонтально, а не вертикально.

Вертикальное масштабирование виртуальных машин имеет свои пределы: вы не можете предоставить больше определенного объема памяти и ЦП. Как только вы попытаетесь выжать максимальную пропускную способность хранилища и полосу пропускания сети, вы начнете испытывать проблемы производительности в других частях вычислительного стека. И это мы еще не учитываем, как столь крупные виртуальные машины ударят по вашему кошельку (или кошельку вашего босса). Хочу напомнить материал главы 9: вертикальное масштабирование показано на рисунке 10.3. А теперь представьте себе кластер таких крупных виртуальных машин баз данных, потому что вам требуется обеспечить избыточность и устойчивость своего приложения, ведь так?

С другой стороны, горизонтальное масштабирование позволяет запускать виртуальные машины баз данных с меньшим объемом ресурсов и, следовательно, меньшими затратами. Это становится возможным благодаря тому, что в базах данных NoSQL данные разделяются между узлами, а поступающие из приложения запросы перенаправляются на соответствующий узел. Узлам в кластере не нужно знать, где хранятся все данные.

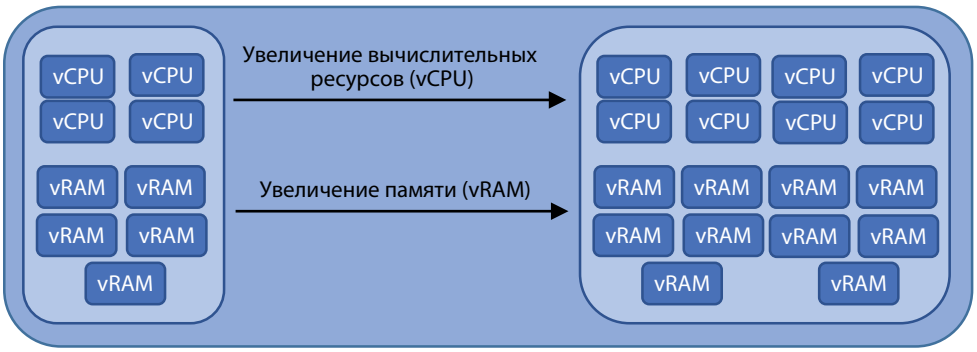


Рис. 10.3. Традиционные структурированные базы данных масштабируются вертикально. По мере роста базы данных увеличиваются объемы хранилища, памяти и мощности ЦП на сервере.

Им просто нужно отвечать на собственные запросы. В ответ на растущий спрос со стороны клиентов можно быстро добавлять узлы в кластер.

Следовательно, нет необходимости умещать всю базу данных NoSQL в памяти хоста. Достаточно хранить и обрабатывать только часть базы данных — *сегмент*. Если ваше приложение обрабатывает большие объемы *структурированных* данных, функционирование базы данных NoSQL может отрицательно сказаться на производительности, потому что разным хостам адресуют запросы на разные фрагменты информации, которые затем возвращают клиенту. Если вам нужно обработать большой объем *неструктурированных* данных, базы данных NoSQL обеспечат не только преимущества с точки зрения управления и эффективности, но и повышенную производительность. Пример горизонтального масштабирования неструктурированных баз данных на уровне узлов показан на рисунке 10.4.

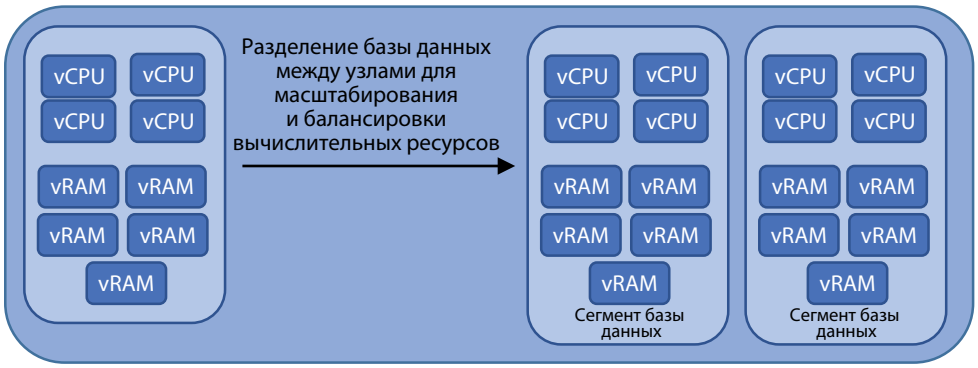


Рис. 10.4. Неструктурированные базы данных NoSQL масштабируются горизонтально. По мере роста базы данных она сегментируется — разделяется на сегменты данных, которые распределяются по всем серверам базы данных.

10.1.4 Cosmos DB — резюме

Что такое Cosmos DB? Это автоматически масштабируемая, глобально распределенная платформа баз данных, которая позволяет использовать разные формы баз данных NoSQL. Как и в случае с такими сервисами, как веб-приложения, Cosmos DB избавляет вас от множества обязанностей в сфере управления. Когда вы создаете веб-приложение, не требуется настраивать балансировку нагрузки или кластеризацию — достаточно выбрать регионы, настроить автомасштабирование, а затем загрузить код своего приложения. Платформа Azure самостоятельно реплицирует и распространяет трафик веб-приложения, соблюдая принципы высокой доступности. Работая с Cosmos DB, вам не нужно беспокоиться о том, насколько большая база данных вам нужна, сколько памяти требуется назначить или как реплицировать данные, чтобы обеспечить избыточность. Вы выбираете, какая пропускная способность вам потребуется и в каких регионах сохранить данные, а затем начинаете добавлять данные.

В этой главе используется SQL-модель Cosmos DB, однако данные хранятся в формате NoSQL — JSON. Возможно, для вас это новые понятия, но постарайтесь следить за тем, что я говорю. Можно использовать и другие модели, в том числе MongoDB, Cassandra, Gremlin и таблицы. Их функциональность одинакова: выберите модель, регионы и начинайте добавлять данные. В этом преимущество Cosmos DB.

10.2 Создание учетной записи и базы данных Cosmos DB

Посмотрим на Cosmos DB и неструктурированные базы данных на практике и узнаем, что можно сделать несколькими способами. Первый способ — использование портала Azure для создания учетной записи, выбор и создание модели базы данных, а также ввод в нее данных, чтобы ваше приложение могло запрашивать их. Либо можно воспользоваться интерфейсом командной строки Azure, Azure PowerShell или наборами SDK для конкретных языков программирования и создать все вышеперечисленное в коде. Давайте воспользуемся порталом Azure, чтобы вы могли видеть, как создаются и запрашиваются ваши данные.

10.2.1 Создание и заполнение базы данных Cosmos DB

В главе 4 вы создали свою первую базу данных NoSQL с таблицей хранилища Azure. Давайте с помощью Cosmos DB создадим аналогичную базу данных, но на этот раз с поддержкой всех вариантов геоизбыточности и репликации, чтобы клиенты могли заказывать пиццу в вашем онлайн-магазине без промедлений. Давайте создадим учетную запись Cosmos DB и базу данных документов, а затем добавим несколько записей данных, соответствующих трем типам пиццы, как показано на рисунке 10.5.

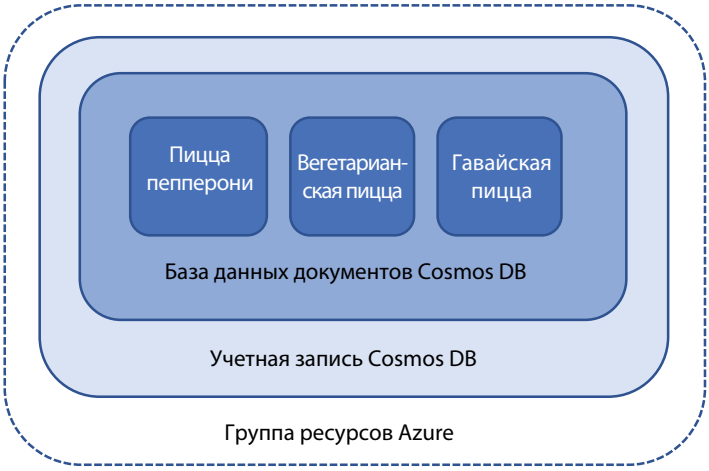


Рис. 10.5. В этом разделе вы создадите группу ресурсов и учетную запись Cosmos DB. Затем в этой учетной записи создается база данных документов, куда добавляются три записи, представляющие базовое меню вашего магазина пиццы.

Попробуйте сейчас

Чтобы увидеть Cosmos DB в действии, создайте учетную запись на портале Azure:

1. Зайдите на портал Azure и нажмите кнопку «Создать ресурс» в верхнем левом углу панели.
2. Найдите и выберите Azure Cosmos DB, а затем нажмите кнопку «Создать».
3. Создайте группу ресурсов, например `azuremolchapter10`, и введите уникальное имя для учетной записи Cosmos DB, такое как `azuremol`.
4. Тип модели, который можно использовать для своей базы данных, называется API. Для этого примера нужно выбрать «Основной (SQL)» из раскрывающегося меню.

5 В строке «Местоположение» выберите «Восток США». Cosmos DB доступна во всех регионах Azure, но в этом примере (а также в веб-приложении, развертываемом в практическом упражнении в конце главы) нужно использовать регион «Восток США».

6 Не включайте параметр геоизбыточности, а также другие дополнительные функции, такие как несколько включенных регионов записи. Вразделе 10.2.2 подробно рассматривается глобальная репликация вашей базы данных.

Защита трафика с помощью конечных точек сервисов

Вы можете подключить Cosmos DB к виртуальной сети Azure с помощью так называемой *конечной точки сервиса*. Мы не будем сейчас обсуждать этот вариант, но это удобная функция, которая защищает экземпляр, разрешая доступ к базе данных только из определенной виртуальной сети.

При создании промежуточных приложений, использующих Cosmos DB, или внутренних приложений можно использовать конечную точку сервиса виртуальной сети, чтобы предоставлять доступ только из определенной виртуальной сети, а не из Интернета и публичной конечной точки. Все больше сервисов Azure поддерживают такие конечные точки. Это еще один пример того, как вы можете защитить среду в соответствии с требованиями бизнеса.

7 Когда все будет готово, просмотрите и создайте учетную запись Cosmos DB. Для этого потребуется несколько минут.

В настоящее время ваша база данных пуста, поэтому давайте поговорим о том, как сохранить базовые данные для меню вашего магазина пиццы. Cosmos DB группирует данные в базе данных в нечто, называемое *контейнером*. Он отличается от контейнеров, которые лежат в основе Docker, Kubernetes и облачных приложений и о которых вы, возможно, слышали. Такая путаница с названиями не идет на пользу, но пока следите за моей мыслью.

Рис. 10.6. База данных Cosmos DB, реализованная по модели документов, хранит данные в коллекциях. Эти коллекции позволяют группировать данные для быстрой индексации и обработки запросов.

В базах данных Cosmos DB, где используется модель документов, данные логически сгруппированы в контейнеры, которые называются *коллекциями*. У других моделей API несколько иное имя для объекта контейнера, например *граф* в Gremlin API. Для нашего SQL API в этих коллекциях хранятся связанные фрагменты данных, которые можно быстро индексировать и запрашивать, как показано на рисунке 10.6. Нельзя сказать, что коллекции совершенно не похожи на то, как организованы традиционные базы данных SQL с помощью таблиц, однако коллекции гарантируют гораздо более высокую гибкость в вопросах распределения данных с целью обеспечения производительности или избыточности.

Поскольку Cosmos DB предназначена для обработки очень больших объемов данных и высокой пропускной способности, вы можете выбрать способ определения размера, а также управления перемещением и стоимостью этих данных. Пропускная способность вычисляется в единицах запроса в секунду (ЕЗ/с), а одна единица запроса эквивалентна 1 КБ данных в документах. В конечном счете вы решаете, какая полоса пропускания должна быть у вашей базы данных. Естественно, чем больше полоса пропускания (ЕЗ/с), тем больше вы платите. Cosmos DB показывает, какой объем данных вы используете и какую пропускную способность использует ваше приложение. Как правило, вам не нужно сильно беспокоиться о подборе оптимальных значений. Что касается вашего магазина пиццы, начнем с малого, не будем сходить с ума!

Попробуйте сейчас

Чтобы создать коллекцию и заполнить базу данных определенным количеством записей, выполните следующие действия, как показано далее:

1. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2. Выберите группу ресурсов, в которой была создана ваша база данных Cosmos DB, например azuremolchapter10.
3. Выберите учетную запись Cosmos DB из списка ресурсов и перейдите на страницу «Обзор».
4. Выберите «Добавить контейнер»
5. Это ваша первая база данных, поэтому выберите «Создать», а затем введите имя, например `pizzadb`.
6. В поле «Пропускная способность» оставьте значение по умолчанию.
7. В качестве идентификатора контейнера введите `pizzas`. В результате создается логический контейнер, который можно использовать для хранения элементов из меню вашего магазина пиццы.
8. Введите ключ раздела `/description`, чтобы убедиться, что типы пиццы распределены равномерно.

   Ключ раздела определяет, как данные могут быть разделены в базе данных. На самом деле он не нужен в небольшом образце базы данных, таком как этот, но его рекомендуется использовать на случай масштабирования приложения.
9. Не выбирайте параметр «Добавить уникальный ключ». Ключи — это способ дальнейшего логического определения контейнера, например выделения подтипов еды, которую могут заказать клиенты. Более широкая коллекция предназначена для целого меню, однако в случае очень больших баз данных вы наверняка захотите использовать ключи разделов для пицц, напитков и десертов.
10. Чтобы создать базу данных и коллекцию, нажмите кнопку «ОК».

Теперь у вас есть учетная запись Cosmos DB, база данных и коллекция, в которой по-прежнему нет ни одной пиццы. Можно импортировать данные или написать код, обеспечивающий ввод пакета данных. Давайте вручную создадим три пиццы, чтобы разобраться с некоторыми графическими инструментами, доступными на портале Azure для просмотра, обработки и запросов данных в вашей базе данных Cosmos DB.

Попробуйте сейчас

Чтобы добавить в базу данных записи, выполните следующие действия, как показано на рисунке 10.7:

1 В своей учетной записи Cosmos DB выберите «Обозреватель данных» в меню в левой части окна «Обзор».

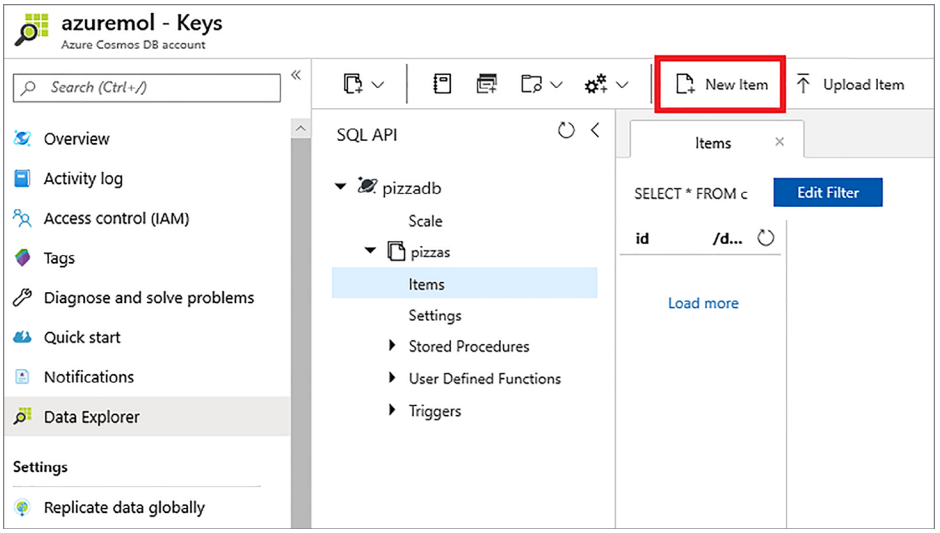


Рис. 10.7. С помощью обозревателя данных на портале Azure можно просматривать свои коллекции, адресовать им запросы и создавать новые документы. С помощью этого графического инструмента можно удобно управлять вашей базой данных из интернет-браузера.

2 Разверните первую базу данных `pizzadb` и коллекцию `pizzas`.

3 Добавьте новый элемент, чтобы поместить несколько пицц в базу данных. Данные добавляются в формате JSON.

4 В текстовом поле замените существующий текст на следующие данные, чтобы создать новый элемент меню для простой пиццы с пепперони:

```
{
    "description": "Pepperoni",
    "cost": "18"
}
```

5 Чтобы добавить данные в базу данных, нажмите «Сохранить».

6 Добавьте в меню еще одну пиццу. На этот раз добавьте свойство, указывающее, что у этой пиццы безглютеновая основа. Нет необходимости совершать какие-либо сложные операции с основной базой данных — достаточно добавить к данным еще одно свойство. Чтобы добавить еще один элемент, введите следующие данные и нажмите кнопку «Сохранить»:

```
{
    "description": "Veggie",
    "cost": "15",
    "gluten": "free"
}
```

7 Добавьте еще один (последний) тип пиццы. На этот раз добавьте свойство, указывающее, какие топпинги могут быть на пицце. Чтобы добавить еще один элемент, введите следующие данные и нажмите кнопку «Сохранить»:

```
{
    "description": "Hawaiian",
    "cost": "12",
    "toppings": "ham, pineapple"
}
```

Эти 3 записи показывают возможности и преимущества базы данных NoSQL. Вы добавили свойства в записи без необходимости обновления схемы базы данных. Два разных свойства указывали на то, что вегетарианская пицца имела безглютеновую основу и какие топпинги могут быть добавлены на гавайскую пиццу. Cosmos DB принимает эти дополнительные свойства и делает соответствующие данные доступными вашим приложениям.

Некоторые дополнительные свойства JSON добавляются для таких элементов, как `id`, `_rid` и `_self`. Сейчас вам не стоит забивать ими голову. Cosmos DB использует эти свойства для отслеживания и идентификации данных. Вам не нужно редактировать или удалять их вручную.

10.2.2 Добавление глобальной избыточности в базу данных Cosmos DB

Теперь у вас есть база данных Cosmos DB, в которой сохранено базовое меню пиццы в регионе «Восток США». Однако ваш магазин пиццы готов открыть франшизы по всему миру! Вам нужно реплицировать данные о своих пиццах в регионы Azure в разных расположениях — ближе к вашим новым клиентам.

Зачем нам это? Если все ваши клиенты начнут считывать данные из базы данных в одном регионе и записывать данные в нее, по подводным кабелям в разные точки мира будет передаваться большой объем трафика. Чтобы обеспечить минимальную задержку для своих клиентов, можно реплицировать данные в регионы Azure в разных точках мира, и тогда клиенты смогут подключаться к ближайшей к ним реплике, как показано на рисунке 10.8.

Модели согласованности и гарантии интегрированы в платформу Cosmos DB, чтобы обеспечить для вас согласованность и репликацию данных. Вы назначаете один или несколько регионов в качестве основного местоположения записи. В рассматриваемых в этой книге примерах используется одна точка записи, однако можно воспользоваться поддержкой нескольких хозяев для записи данных в ближайшую конечную точку, которая затем асинхронно распространяется в другие регионы. Кроме того, данные быстро реплицируются в назначенные вами регионы чтения. Можно управлять порядком отработки отказа, назначать регионы чтения и автоматически или вручную задавать регионы, из которых приложение будет выполнять чтение.

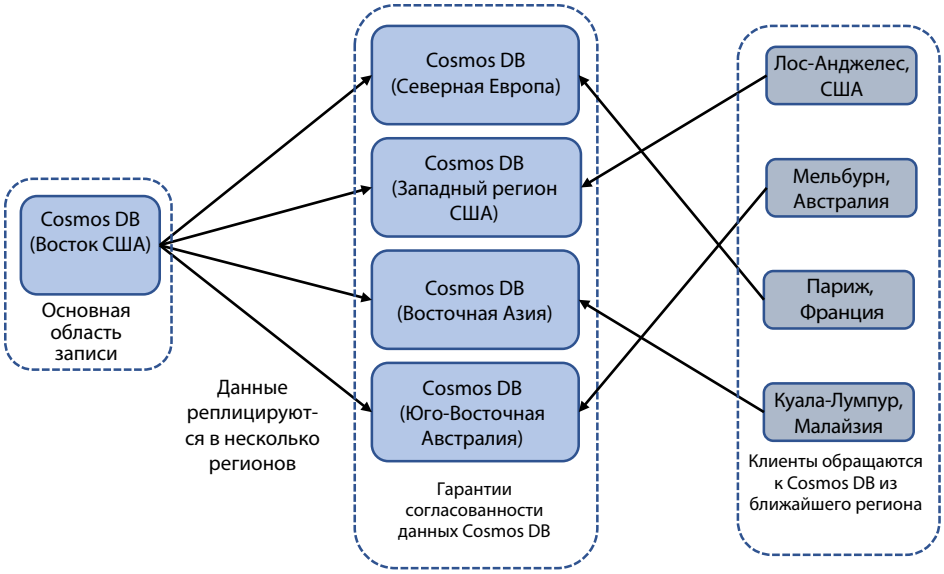


Рис. 10.8. Данные реплицируются из одного основного экземпляра Cosmos DB в несколько регионов Azure по всему миру. Можно отправить веб-приложениям инструкцию выполнять считывание из ближайшего региона и наладить динамическую маршрутизацию клиентов в ближайшее к ним расположение — это обеспечит минимальную задержку и улучшит показатели времени ответа.

Можно определить модель согласованности (это, скорее, вопрос проектирования, а не операционный аспект), которая определяет скорость репликации операций записи в разных регионах. Модели согласованности варьируются от *строгих* (система ждет подтверждения реплицированных операций записи репликами и тем самым гарантирует согласованность операций чтения) до *случайных*, не таких строгих. Эти модели гарантируют, что все данные реплицируются, однако возможны небольшие задержки, когда операции чтения из реплик возвращают разные значения. Задержка длится до тех пор, пока все данные не будут синхронизированы.

Существует баланс между ограниченным географическим распределением (строгая модель согласованности) и широкой географической репликацией со случайной моделью согласованности, однако необходимо понимать, что репликация данных вызывает небольшую задержку. Нельзя упускать из вида и затраты на полосу пропускания и обработку данных — они зависят от того, насколько согласованно и быстро требуется реплицировать данные. Платформа Azure выполняет базовую репликацию данных из точки записи; не требуется создавать приложения для репликации данных или определения оптимального способа считывания данных с реплицированных конечных точек.

В глобальных масштабах это означает возможность использования нескольких виртуальных машин или веб-приложений, созданных вами в предыдущих главах, но в разных регионах мира. Эти приложения подключаются к локальному экземпляру Cosmos DB для запроса и чтения всех данных. Благодаря

некоторым полезным функциям Azure для работы с сетевым трафиком, которые мы рассмотрим в главе 11, пользователи могут автоматически перенаправляться в один из локальных экземпляров веб-приложения, которые также используют локальный экземпляр Cosmos DB. В случае региональных отключений или обслуживания вся платформа перенаправляет клиента в следующий ближайший к нему экземпляр.

В мире традиционных структурированных баз данных, где необходимо управлять виртуальными машинами, устанавливать базы данных и конфигурировать кластеры, подобная настройка требует серьезного планирования и сложна в реализации. С Cosmos DB для этого достаточно 3 щелчков мыши. Честное слово!

Попробуйте сейчас

Чтобы реплицировать свои данные Cosmos DB в глобальных масштабах, выполните следующие действия:

1. Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2. Выберите группу ресурсов, в которой была создана ваша база данных Cosmos DB, например `azuremolchapter10`.
3. Выберите свою учетную запись Cosmos DB из списка ресурсов. Эти 2 щелчка мыши вам ничего не стоили, но можно начинать считать!
4. Выберите пункт меню слева, чтобы реплицировать данные глобально. На карте со всеми доступными регионами Azure видно, что в настоящее время ваша база данных доступна в регионе «Восток США» (см. рис. 10.9).

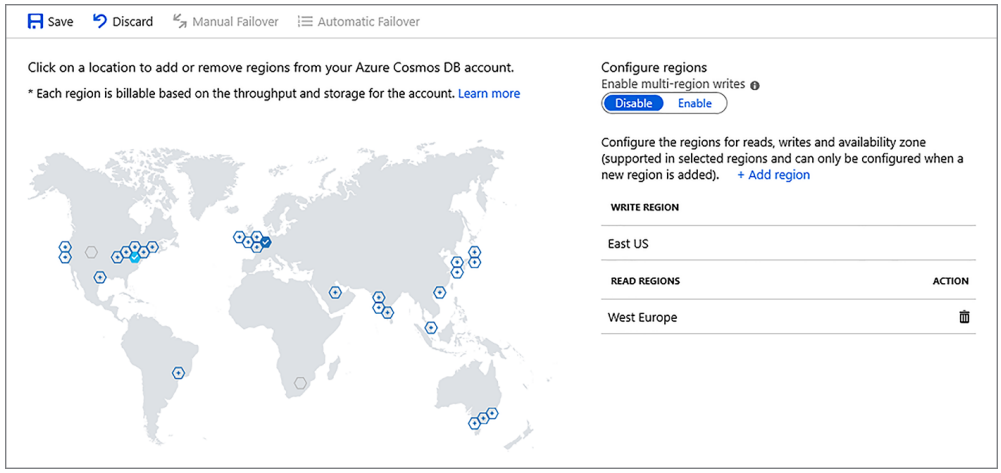


Рис. 10.9. Выберите регион Azure, чтобы реплицировать в него свою базу данных Cosmos DB, а затем нажмите «Сохранить». Это все действия, необходимые для глобального распределения ваших данных.

5 Выберите «Западная Европа» и нажмите «Сохранить». Можно выбрать любой регион Azure, однако в практическом упражнении в конце главы указано, что ваши данные необходимо реплицировать в Западную Европу. Всего через несколько минут данные будут реплицированы в выбранный регион и станут доступны онлайн для использования вашими приложениями.

Отлично, вы считаете щелчки мыши? Три щелчка, ведь так? Давайте будем щедрыми и учтем первые два щелчка мыши, которые потребовались, чтобы выбрать группу ресурсов и учетную запись Cosmos DB. Итак, не больше 5 щелчков мыши и всего несколько секунд — и готов реплицированный экземпляр вашей базы данных, благодаря чему ваши приложения могут осуществлять доступ к данным из ближайшего к ним региона. Возможно ли такое с традиционным кластером MySQL? Напишите мне в Твиттере по адресу @fouldsy, если у вас получится сделать это так же быстро за пределами Cosmos DB!

Теперь, когда ваша база данных распределена по всему миру, нужно ли вносить множество изменений в код, чтобы определить, к какому региону Cosmos DB подключаться? Как обслуживать все эти версии ваших приложений с учетом региона Azure, в котором они выполняются? Легко! Платформа Azure все сделает за вас!

10.3 Доступ к глобально распределенным данным

В большинстве своем платформа Azure определяет оптимальное расположение для взаимодействия с вашим приложением. Как правило, приложению нужно читать и записывать данные. Можно определить для базы данных Cosmos DB политики отработки отказа, которые будут определять основное расположение записи. Это расположение записи функционирует в качестве центрального концентратора, который гарантирует, что данные согласованно реплицируются по регионам. Однако ваше веб-приложение, как правило, может выполнять чтение из нескольких доступных регионов, чтобы ускорить обработку запросов и вернуть данные клиенту. Все это обеспечивается вызовами REST.

Посмотрим, что происходит в интерфейсе командной строки Azure, когда вы запрашиваете информацию о базе данных Cosmos DB. Это можно сравнить с приложением, которое подключается к базе данных, но не дает вам углубиться в код слишком сильно.

Попробуйте сейчас

Используйте команду `az cosmosdb show`, чтобы найти информацию о расположениях чтения и записи.

1 Зайдите на портал Azure в браузере и откройте Cloud Shell.

2 Используйте команду `az cosmosdb show`, чтобы просмотреть расположения чтения и записи для своей базы данных Cosmos DB.

Введите имя группы ресурсов и имя базы данных, которые вы создали в предыдущих упражнениях «Попробовать». В следующем примере рассматриваются группа ресурсов `azuremolchapter10` и база данных Cosmos DB с именем `azuremol`:

```
az cosmosdb show \
  --resource-group azuremolchapter10 \
  --name azuremol
```

Эта команда возвращает большой объем выходных данных, поэтому давайте рассмотрим два ключевых аспекта: расположения чтения и расположения записи. Вот несколько примеров выходных данных для раздела `readLocations`:

```
"readLocations": [
   {
     "documentEndpoint":"https://azuremol-eastus.documents.azure.com:443/",
     "failoverPriority": 0,
     "id": "azuremol-eastus",
     "isZoneRedundant": "false",
     "locationName": "East US",
     "provisioningState": "Succeeded"
   },
   {
     "documentEndpoint":
      "https://azuremol-westeurope.documents.azure.com:443/",
     "failoverPriority": 1,
     "id": "azuremol-westeurope",
     "isZoneRedundant": "false",
     "locationName": "West Europe",
     "provisioningState": "Succeeded"
     }
],
```

Когда приложение подключается к базе данных Cosmos DB, можно задать политику подключения. Если вы не привыкли работать с базами данных, представьте базовое подключение ODBC, которое можно настроить на компьютере с Windows. В строке подключения, как правило, определены имя хоста, имя базы данных, порт и учетные данные. Cosmos DB ничем не отличается от этого. К базе данных Cosmos DB можно подключаться на разных языках, включая .NET, Python, Node.js и Java. Языки могут различаться, но во всех наборах SDK реализована общая функция: обнаружение конечных точек. Важны два основных свойства политики подключения:

- *Автоматическое обнаружение конечных точек* — набор SDK считывает все доступные конечные точки из Cosmos DB и использует указанный порядок отработки отказа. Такой подход гарантирует, что ваше приложение всегда следует порядку, заданному вами на уровне базы данных. Возможно, вы хотите, чтобы все операции считывания выполнялись в регионе «Восток США», а регион «Западная Европа» использовался только при проведении обслуживания в основном расположении.
- *Предпочтительные расположения конечных точек* — вы указываете расположения, которые требуется использовать. Например, если приложение развернуто в Западной Европе, вы, возможно, хотите использовать конечную точку «Западная Европа». В этом случае вы утратите определенную гибкость в добавлении или удалении конечных точек, но гарантируете, что конечная точка по умолчанию расположена близко к вашему приложению, и вам не потребуется использовать сложную сетевую маршрутизацию, чтобы определить нужные параметры конфигурации.

Как правило, ваше приложение делегирует все эти обязанности пакету SDK Cosmos DB. Ваше приложение не меняет способ установки подключения к базе данных: оно просто «знает», что существуют разные расположения, к которым приложение *может* подключиться. Однако именно набор SDK *устанавливает* подключение и использует осведомленность о расположении.

На рисунке 10.10 в упрощенном виде изображено, как осведомленность о расположении используется между вашим приложением и SDK. Повторюсь: язык не имеет значения — подход одинаков: на рисунке используется набор SDK языка Python, потому что именно на нем были написаны пара примеров. Кроме того, в этом примере исходят из того, что вы используете автоматические расположения конечных точек.

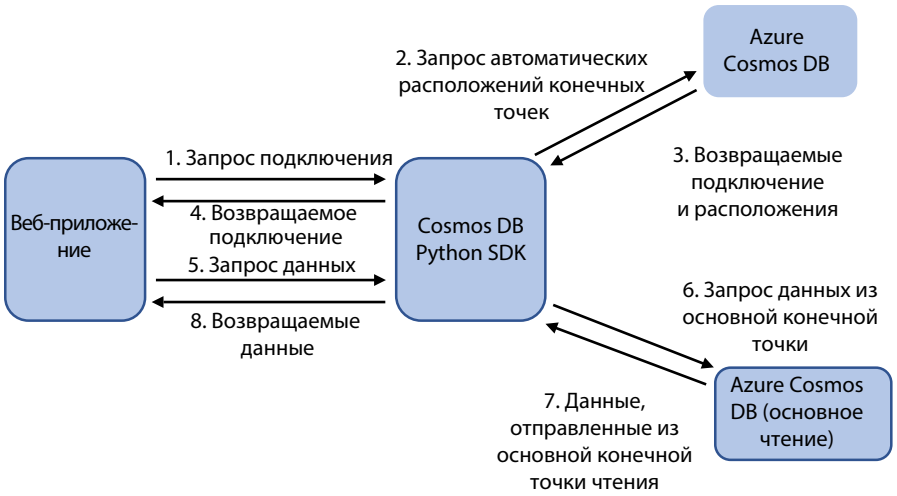


Рис. 10.10. Поток запросов через SDK Cosmos DB, когда приложение использует осведомленность о расположении для отправки запросов в Cosmos DB

На рисунке 10.10 показаны следующие шаги:

1. Вашему приложению необходимо подключиться к базе данных Cosmos DB. В политике подключений включено автоматическое обнаружение конечных точек. Приложение использует SDK Cosmos DB для подключения к базе данных.
2. SDK Cosmos DB отправляет запрос на подключение и указывает, что требуется использовать автоматические расположения конечных точек.
3. Подключение возвращается с учетом запрошенных учетных данных и базы данных.
4. SDK возвращает объект подключения, которое должно использовать приложение. Сведения о расположении абстрагируются из приложения.
5. Приложение запрашивает определенные данные из базы данных Cosmos DB. SDK снова используется для запроса и получения данных.
6. SDK использует список доступных конечных точек и запрашивает первую доступную конечную точку. Затем SDK использует конечную точку подключения для запроса данных. Если основная конечная точка недоступна, например во время обслуживания, автоматически используется расположение следующей конечной точки.

7 Cosmos DB возвращает данные из расположения конечной точки.
8 SDK возвращает данные из Cosmos DB обратно в приложение для синтаксического разбора и отображения должным образом.

Последнее, на что следует обратить внимание в Cosmos DB, — это ключи доступа. Они позволяют контролировать, кто может получить доступ к данным и какие разрешения у них есть. Ключи можно воссоздавать; и, как и в случае с паролями, можно внедрить политику регулярного воссоздания ключей. Чтобы получить доступ к распределенным данным в Cosmos DB, нужно получить свои ключи. На портале Azure можно просмотреть все ключи и строки подключения для своей базы данных.

Попробуйте сейчас

Чтобы просмотреть ключи для своей учетной записи Cosmos DB, выполните следующие действия:

1 Найдите и нажмите кнопку «Группа ресурсов» на панели навигации слева на портале Azure.
2 Выберите группу ресурсов, в которой была создана ваша база данных Cosmos DB, например `azuremolchapter10`.
3 Выберите свою учетную запись Cosmos DB из списка ресурсов.
4 Выберите «Ключи» в левой части экрана.
5 Запомните идентификатор URI и основной ключ (рис. 10.11). Эти значения потребуются вам при выполнении практического упражнения в конце главы.

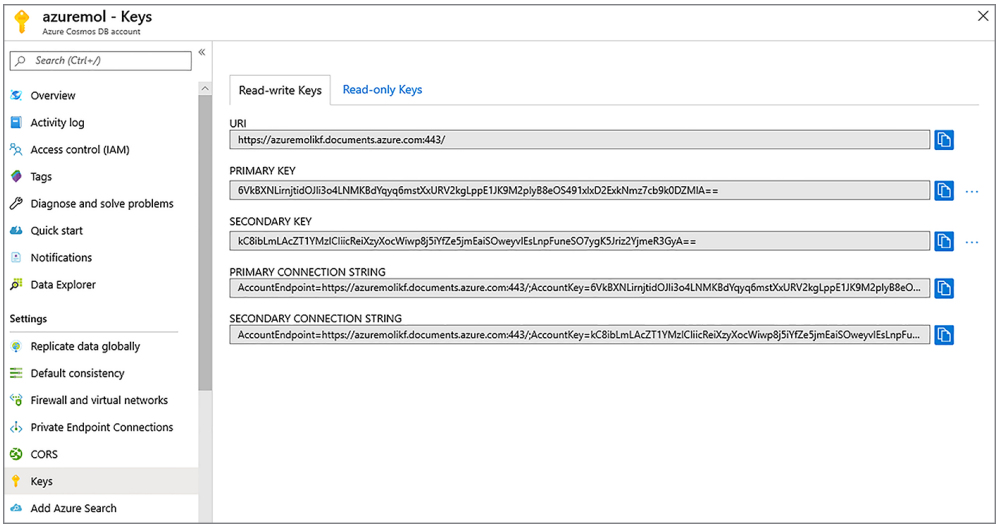


Рис. 10.11. В разделе «Ключи» вашей учетной записи Cosmos DB приводится информация о подключении и ключах доступа. Эта информация потребуется вам при разработке и запуске приложений, например при выполнении практического упражнения в конце главы.

Очень многое в Cosmos DB происходит «под капотом», то есть незаметно для вас: ваши данные распределяются, а ваши приложения получают доступ на чтение и запись в наиболее подходящих расположениях. Но в этом вся суть. Знание о том, что делает для вас сервис Cosmos DB, помогает проектировать и планировать приложение, а также диагностировать и устранять неполадки, если приложения не позволяют SDK выполнять операции чтения и записи, когда необходимо. Вам не нужно беспокоиться о том, когда и как это будет происходить: сконцентрируйтесь на своих приложениях и пользуйтесь облачной функциональностью и преимуществами сервисов Azure, таких как Cosmos DB, чтобы наладить операционную деятельность в глобальных масштабах.

10.4 Практическое упражнение: развертывание веб-приложения, использующего Cosmos DB

В разделе 10.2.2 вы занимались глобальным распределением своей базы данных Cosmos DB. Затем вы получили массу теоретических сведений о том, как веб-приложения могут считывать данные из расположений по всему миру. А сейчас вы наверняка хотите посмотреть, как все это происходит на практике! В этом упражнении используется базовое веб-приложение из предыдущих глав, но на этот раз меню пиццы состоит из элементов, добавленных в базу данных Cosmos DB из упражнения «Попробовать»:

1. Создайте веб-приложение на портале Azure.
2. Так как магазин пиццы — это больше не базовая HTML-страница, выберите `Node LTS` в качестве среды выполнения, которая работает в Linux.
3. Когда веб-приложение будет готово, создайте источник развертывания (локальный репозиторий Git). Необходимо выполнить те же действия, что и при создании объектов в предыдущих главах (например, в главе 3), поэтому если вам нужно освежить знания, обратитесь к тем упражнениям.
4. Откройте Cloud Shell. В предыдущих главах вы получили копию примеров для Azure из GitHub. Если нет, копию можно получить так:

```
git clone https://github.com/foulDsy/azure-mol-samples-2nd-ed.git
```

5. Перейдите в каталог, который содержит пример веб-приложения Cosmos DB:

```
cd ~/azure-mol-samples-2nd-ed/10/cosmosdbwebapp
```

6. Отредактируйте файл конфигурации с URI и ключом доступа к базе данных, которые вы копировали в предыдущем упражнении «Попробовать», чтобы просмотреть свои ключи Cosmos DB:

```
nano config.js
```

7. Запишите файл, нажав клавиши CTRL+O, а затем выйдите из системы, нажав клавиши CTRL+X.
8. Добавьте элементы и подтвердите изменения в Git, выполнив следующую команду:

```
git init && git add . && git commit -m "Pizza"
```

9 Создайте ссылку на новый репозиторий Git в промежуточном слоте с помощью команды `git remote add azure` и укажите после нее URL-адрес развертывания Git.

10 С помощью команды `git push azure master` отправьте изменения в ваше веб-приложение.

11 Выберите URL-адрес своего веб-приложения в окне «Обзор» на портале Azure.

12 Откройте этот URL-адрес в веб-браузере, чтобы просмотреть свой магазин пиццы, который теперь функционирует на базе Cosmos DB, как показано на рисунке 10.12.

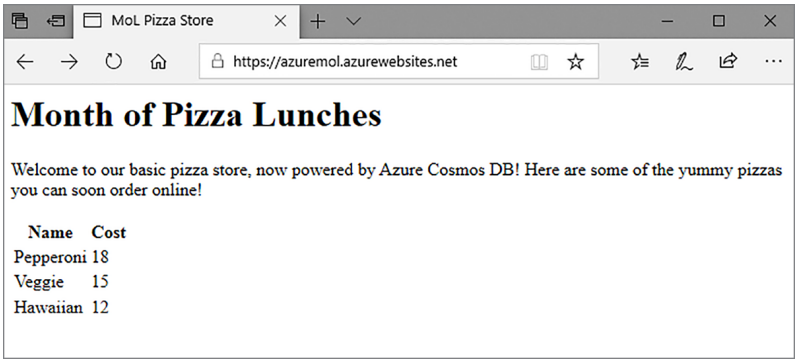


Рис. 10.12. В основном веб-приложении Azure отображается небольшое меню пиццы с данными, которые вы ввели в базе данных Cosmos DB. Отображается пиццерия из предыдущих глав, но теперь список пицц и их цены основаны на данных из Cosmos DB. Сайт по-прежнему базовый, так как цель состоит в том, чтобы вы увидели сервис в действии и поняли, как создавать собственные приложения.

11 Управление сетевым трафиком и маршрутизацией

Разрешение сервиса доменных имен (DNS) лежит в основе практически всех устанавливаемых вами цифровых подключений. Именно с помощью этой технологии вы просматриваете веб-страницы, получаете сообщения электронной почты, смотрите сериалы на Netflix и звоните по Skype. DNS — это механизм, который преобразует имя, например manning.com, в IP-адрес. Если мне нужно узнать что-то новое, мне не нужно запоминать IP-адрес 35.166.24.88 — достаточно ввести в браузере `manning.com` и найти нужные книги! Сетевые устройства маршрутизируют трафик по IP-адресам, поэтому нужно средство, которое помогало бы нам — пользователям с плохой памятью — покупать книги и заказывать пиццу в Интернете.

В предыдущих главах мы много говорили о создании масштабируемых высокодоступных приложений с возможностью глобального распределения. Осталось обсудить только то, как направлять клиентов из разных стран мира, в наиболее подходящие экземпляры приложения (как правило, экземпляры, которые находятся к ним ближе всего). Диспетчер трафика Azure упрощает автоматическое направление клиентов в подходящие экземпляры приложения на основе производительности или географического расположения. В этой главе мы рассмотрим, как создавать зоны DNS и управлять ими в Azure, а также как использовать диспетчер трафика для направления клиентов с запросами DNS, как показано на рисунке 11.1.

11.1 Что такое Azure DNS?

Для прохождения этой главы и использования Azure DNS не требуется глубокого понимания принципов работы DNS. На рисунке 11.2 представлены общие обзорные сведения о том, как пользователь отправляет запрос в сервис DNS, чтобы получить IP-адрес веб-приложения. Шаги 1 и 2 могут состоять из множества подшагов, поэтому если в конце этой главы у вас останется немного времени, рекомендую вам прочитать о том, как функционируют запросы DNS и рекурсия.

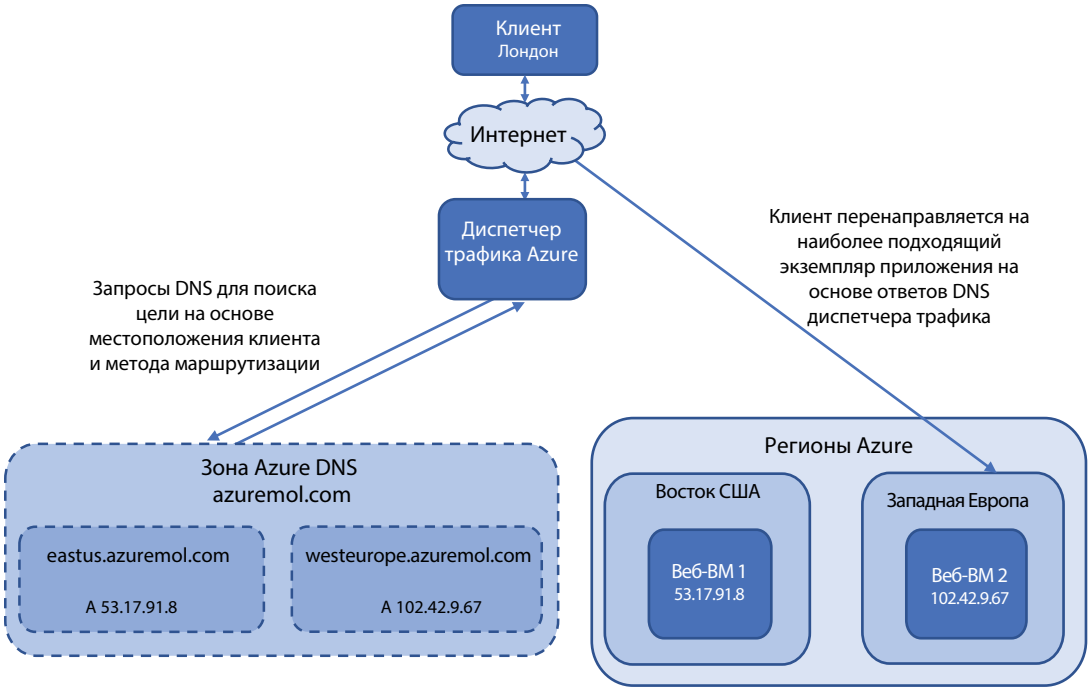


Рис. 11.1. В этой главе мы рассмотрим создание зон DNS в Azure DNS. Чтобы свести к минимуму задержку и улучшить показатели времени отклика, диспетчер трафика может использоваться для отправки запросов DNS и направления клиентов в ближайший к ним экземпляр приложения.

Azure DNS функционирует так же, как любое другое знакомое вам решение DNS. Возможно, вы даже пользуетесь одним из них. Ваша зона и записи хранятся в Azure, а серверы имен, отвечающие на запросы DNS, глобально распределены по центрам обработки данных Azure.

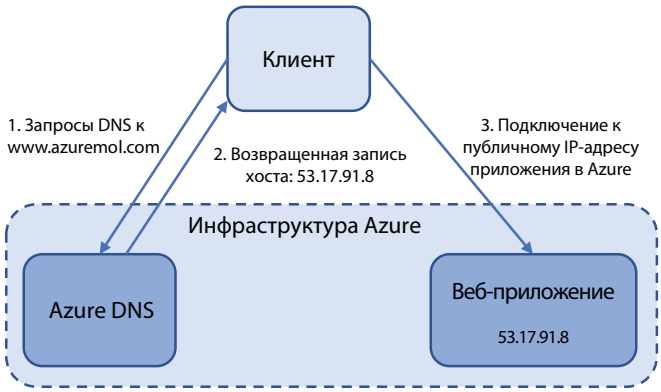


Рис. 11.2. Упрощенное изображение трафика DNS показывает, как пользователь отправляет DNS-запрос адреса www.azuremol.com на сервер DNS, получает ответ, который содержит связанный IP-адрес, а затем подключается к веб-приложению.

Azure DNS поддерживает все типы записей, которые можно встретить в стандартном сервисе DNS. Можно создать записи типа IPv4 и IPv6. Доступны следующие типы записей:

- *A* — записи хоста IPv4, которые указывают клиентам на ваши приложения и сервисы
- *AAAA* — записи хоста IPv6 — для тех «крутышек», которые используют IPv6, чтобы указать клиентам на ваши приложения и сервисы
- *CNAME* — записи канонического имени или псевдонима (например, сокращенное имя, которое удобнее использовать, чем полное имя хоста сервера)
- *MX* — записи обмена почтой, направляющие трафик электронной почты на ваши почтовые серверы или вашему поставщику
- *NS* — записи сервера имен, которые включают автоматически созданные записи для серверов имен Azure
- *PTR* — записи указателя, позволяющие обратным запросам DNS сопоставлять IP-адреса именам хостов
- *SOA* — это начальные записи зоны, которые включают автоматически созданные записи для серверов имен Azure
- *SRV* — записи сервиса, обеспечивающие обнаружение сетевых сервисов, например для идентификаторов
- *TXT* — текстовые записи, например для Sender Protection Framework (SPF) или DomainKeys Identified Mail (DKIM)

В стандартной конфигурации DNS настраивается несколько серверов DNS. Даже если реализовано географическое распределение этих серверов с целью избыточности, клиенты могут отправлять запросы на сервер имен, который находится на другой стороне земного шара. На отправку запроса, разрешение и запрос ответа для веб-приложения требуются миллисекунды, однако если на сайте много людей, которые хотят заказать пиццу, работа приложения может замедлиться.

Зона Azure DNS реплицируется глобально в нескольких центрах обработки данных Azure. *Сеть произвольной рассылки* гарантирует, что когда клиент отправляет DNS-запрос в ваш домен, на этот запрос отвечает ближайший доступный сервер имен. Как маршрутизации произвольной рассылки это удается? Как правило, один IP-адрес объявляется в нескольких регионах. Вместо того чтобы отправлять простой DNS-запрос, который разрешается в единственный IP-адрес, существующий только в одном расположении, маршрутизация произвольной рассылки позволяет сетевой инфраструктуре интеллектуально определять источник запроса и направлять клиента в ближайший регион с объявленным IP-адресом. Такая маршрутизация позволяет клиентам быстрее подключаться к вашему веб-приложению и в целом улучшает взаимодействие с клиентами.

Не нужно быть экспертом по сетевым технологиям, чтобы разобраться в том, как это работает — Azure все сделает за вас! Объединяя Azure DNS с диспетчером трафика Azure (раздел 11.2.), вы не только возвращаете DNS-запросы с ближайших серверов имен, но и подключаете клиентов к ближайшему к ним экземпляру приложения. Каждая миллисекунда на счету!

11.2 Делегирование реального домена сервиса Azure DNS

Когда вы регистрируете реальный домен, провайдер предоставляет вам интерфейс управления и инструменты для управления этим доменом. Чтобы предоставить клиентам доступ к вашим сервисам и право использования зоны и записей Azure DNS, следует делегировать полномочия на выполнение операций со своим доменом серверам

имен Azure. В этом случае все DNS-запросы будут немедленно направляться на серверы имен Azure, как показано на рисунке 11.3. В настоящее время Azure не позволяет приобретать и регистрировать домены на своей платформе, так что вам потребуется приобрести доменное имя через внешнего регистратора, а затем указать серверам имен Azure на записи сервера имен.

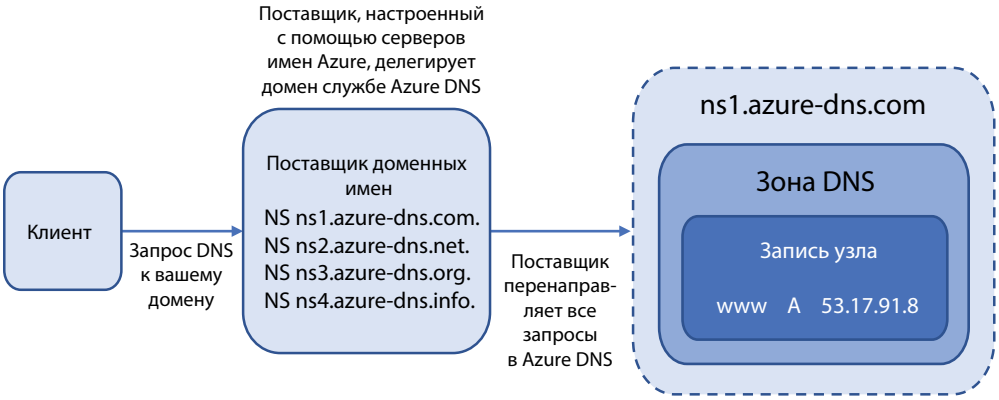


Рис. 11.3. Чтобы делегировать свой домен Azure, настройте текущего поставщика доменов, указав сервер имен и адреса Azure. Когда клиент адресует DNS-запрос вашему домену, запросы отправляются непосредственно на серверы имен Azure в вашей зоне.

Зачем делегировать свой сервис DNS платформе Azure? Чтобы упростить управление и операции. Если вы создаете дополнительные сервисы, корректируете конфигурацию балансировщика нагрузки или хотите улучшить время отклика службы DNS с глобальной репликацией, Azure предоставляет единый интерфейс управления для выполнения этих задач. Если зоны DNS размещены в Azure, можно также реализовать некоторые функции безопасности диспетчера ресурсов, которые мы рассматривали в главе 6: речь идет о таких функциях, как управление доступом на основе ролей (RBAC) с целью ограничения и аудита доступа в зоны DNS и блокировки ресурсов во избежание случайного или умышленного удаления зон.

Большинство регистраторов доменных имен предоставляют лишь базовые интерфейсы и инструменты для управления зонами и записями DNS. Чтобы сократить затраты на управление и повысить безопасность, Azure DNS позволяет использовать для добавления и редактирования записей интерфейс командной строки Azure, Azure PowerShell и API-интерфейсы REST. Те же самые инструменты и рабочие потоки могут использовать операционные специалисты для внедрения новых сервисов; и если возникают проблемы, диагностику и устранение неполадок выполнить проще, если можно не сомневаться в том, что DNS функционирует должным образом. В этом случае из уравнения исчезает одно неизвестное — сторонний поставщик DNS.

Итак, если вы убедились в целесообразности делегирования своего домена сервису Azure DNS, на какие серверы имен Azure следует указать своему домену? Когда вы создаете зону Azure DNS, соответствующие серверы имен перечислены на портале, как показано на рисунке 11.4. Для доступа к этим серверам имен (и их адресам) можно также использовать интерфейс командной строки Azure и Azure PowerShell.

На последних страницах этого раздела нет упражнений «Попробовать», потому что чтобы протестировать маршрутизацию трафика, необходимо приобрести и настроить реальный домен. Можно создать зону Azure DNS без реального домена,

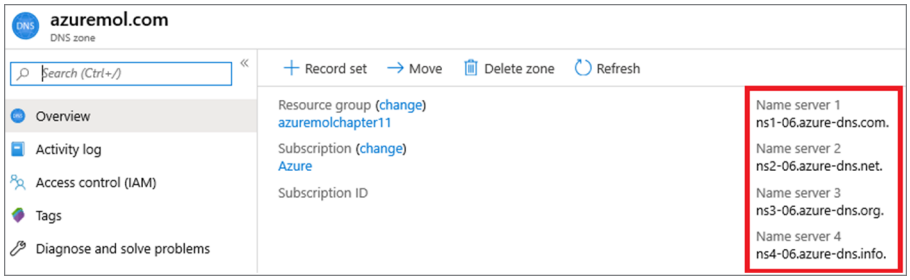


Рис. 11.4. Вы можете просмотреть серверы имен Azure для своей зоны DNS на портале Azure, с помощью интерфейса командной строки Azure или Azure PowerShell.

но на него не будет направляться трафик. В реальной жизни вы обновляете записи на сервере имен, а ваш поставщик направляет любые запросы, адресованные вашему домену, на серверы имен Azure. Чтобы делегирование вашего домена распространилось по глобальной иерархии DNS, должно пройти от 24 до 48 часов (обычно, однако, гораздо меньше), так что планируйте развертывание соответственно, поскольку возможны кратковременные перерывы в работе вашего приложения.

11.3 Глобальная маршрутизация и разрешение адресов с помощью диспетчера трафика

В предыдущих главах вы узнали о глобально распределенных высокодоступных приложениях. Конечная цель — это наличие нескольких экземпляров веб-приложения или виртуальной машины в разных регионах или на разных континентах, которые подключаются к ближайшему к ним экземпляру Cosmos DB. Но как сделать так, чтобы ваши клиенты подключались к ближайшей виртуальной машине или веб-приложению, в котором выполняется ваше приложение?

Диспетчер трафика Azure — это сетевой сервис, который функционирует как центральное место назначения для ваших клиентов. В качестве примера рассмотрим веб-приложение по адресу www.azuremol.com. На рисунке 11.5 представлен обзор того, как диспетчер трафика перенаправляет пользователей в ближайшее доступное приложение.

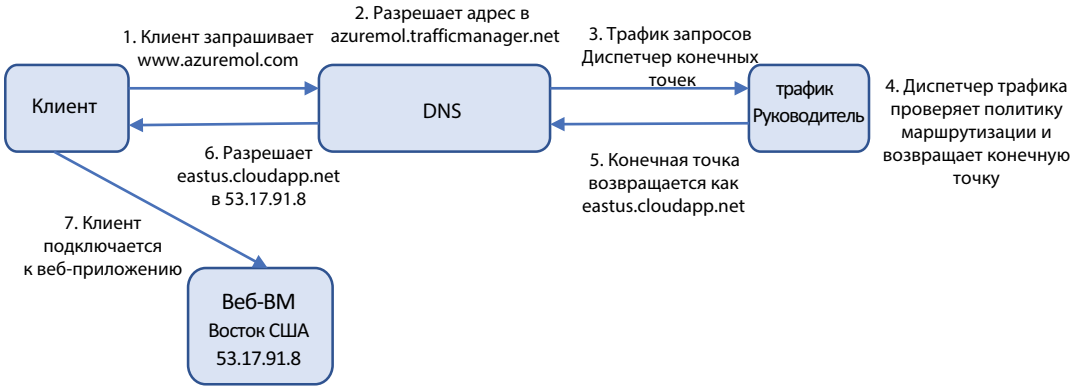


Рис. 11.5. Клиент отправляет в сервис DNS запрос адреса www.azuremol.com. Служба DNS перенаправляет запрос диспетчеру трафика, который возвращает конечную точку с учетом используемого метода маршрутизации. Конечная точка разрешается в IP-адрес, который клиент использует для подключения к веб-приложению.

Диспетчер трафика не выполняет роль балансировщика нагрузки, о которой говорилось в главе 8. Как показано на рисунке 11.5, диспетчер трафика перенаправляет трафик на публичный IP-адрес. Рассмотрим поток трафика внимательнее:

1. Пользователь отправляет DNS-запрос адреса www.azuremol.com. Соответствующий сервер DNS обращается к серверам имен по поводу адреса azuremol.com (если вы используете Azure DNS, это, возможно, серверы имен Azure) и запрашивает запись для www.
2. Www-хост выполняет разрешение в запись CNAME, которая указывает на адрес azuremol.trafficmanager.net.
3. Сервис DNS перенаправляет DNS-запрос на trafficmanager.net серверам имен Azure.
4. Затем диспетчер трафика рассматривает запрос и определяет конечную точку, на которую требуется направить пользователя. Рассматриваются работоспособность и состояние конечной точки — как в случае с балансировщиками нагрузки Azure. Кроме того, анализируется метод маршрутизации диспетчера трафика. Диспетчер трафика может использовать следующие методы маршрутизации:
   - *По приоритету* — контролирует порядок доступа к конечным точкам
   - *Взвешенный* — трафик распределяется между конечными точками на основании назначенного показателя веса
   - *По производительности* — маршрутизация пользователей по конечным точкам с учетом задержки, чтобы пользователю было обеспечено минимально возможное время отклика
   - *По географическому принципу* — конечные точки связываются с географическим регионом, а пользователи перенаправляются к ним с учетом местоположения
5. Диспетчер трафика возвращает сервису DNS конечную точку eastus.cloudapp.net.
6. Сервис DNS ищет DNS-запись для eastus.cloudapp.net и возвращает результат запроса клиенту.
7. Получив IP-адрес запрошенной конечной точки, клиент обращается к веб-приложению напрямую. На этом этапе трафик может попасть на публичный IP-адрес балансировщика нагрузки Azure, а не непосредственно на виртуальную машину.

Как видите, задача диспетчера трафика — определить конечную точку заданного приложения, чтобы направить туда клиентов. Существует несколько проверок, позволяющих отслеживать состояние конечных точек (аналогичных проверкам работоспособности балансировщика нагрузки, о которых вы узнали в 8 главе). Кроме того, можно определить взвешенный механизм маршрутизации трафика или механизм маршрутизации по приоритету, чтобы распределить пользователей по набору доступных конечных точек (повторюсь, эта процедура схожа с работой балансировщика нагрузки). Как правило, диспетчер трафика направляет трафик на балансировщик нагрузки Azure, в шлюз приложений или в развернутое веб-приложение.

Azure Front Door

Диспетчер трафика, который мы рассмотрим в этом разделе, отлично подходит для глобального распределения и маршрутизации трафика. Он работает с любым типом конечной точки Интернета, а не только с ресурсами в Azure. Маршрутизация трафика основана на DNS и не зависит от самого приложения.

Если вам необходимы распределение трафика на уровне приложений и возможность разгрузки TLS/SSL или маршрутизации запросов HTTP/HTTPS, вам поможет сервис Azure Front Door. Диспетчер трафика

(продолжение)

и Front Door предлагают одинаковые тип сервиса и параметры конфигурации, но Front Door разработан специально для работы на уровне приложений. Front Door также предоставляет ряд интересных функций для повышения производительности, такие как разделение TCP на мелкие соединения для уменьшения задержки.

Еще в главе 8 мы рассмотрели подсистемы балансировки нагрузки и шлюз приложений, который работает на уровне приложений и выполняет такие задачи, как разгрузка TLS. Основное внимание в этой главе было уделено подсистемам балансировки нагрузки, чтобы вы могли понять основные концепции, на которых будет основываться шлюз приложений. То же применимо и здесь. В этой главе мы сосредоточимся на диспетчере трафика, хотя многие из тех же концепций и параметров конфигурации, таких как параметры маршрутизации, также доступны для Azure Front Door. Как и в большинстве случаев в Azure, возможности каждого сервиса, которые вы используете, зависят от запущенных приложений и их потребностей.

11.3.1 Создание профилей диспетчера трафика

Диспетчер трафика с помощью профилей определяет, какой метод маршрутизации следует использовать и каковы связанные конечные точки для соответствующего запроса. В продолжение начатой в предыдущих главах темы о глобально распределенном приложении напомню, что наша цель — сделать так, чтобы пользователи направлялись в ближайшее к ним веб-приложение. Если мы посмотрим на методы маршрутизации еще раз, то поймем, что обеспечить это можно двумя способами:

- *Маршрутизация по производительности* — клиент направляется в конечную точку с самой низкой задержкой относительно источника запроса. Этот метод маршрутизации подразумевает использование аналитики в определенном объеме и позволяет диспетчеру трафика всегда направлять клиента в доступную конечную точку.
- *Географическая маршрутизация* — клиент всегда направляется в определенную конечную точку на основе источника запроса. Если клиент находится в Соединенных Штатах, он всегда направляется в регион «Восток США», например. Для использования этого метода маршрутизации нужно определить географические регионы, которые будут связаны с каждой конечной точкой.

При использовании географической маршрутизации вы получаете больший контроль над тем, какие конечные точки используют ваши клиенты. Согласно действующему законодательству, клиенты из государственного сектора в определенном регионе могут быть обязаны всегда использовать конечные точки в том же регионе. В упражнениях географические конечные точки используются для реалистичности изложения, потому что хитрость географической маршрутизации в том, что необходимо задать *дочерний профиль*, а не конечную точку напрямую.

Мир не рухнет, если вы будете использовать метод географической маршрутизации с конечными точками, однако для передачи трафика на финальную конечную точку рекомендуется использовать другой профиль диспетчера трафика. Зачем? Регионы могут быть связаны только с одним профилем диспетчера трафика. В предыдущих главах,

посвященных высокой доступности, мы всегда говорили о необходимости обеспечить избыточность. Если связать регион с заданной конечной точкой и использовать географическую маршрутизацию, у вас не останется варианта отработки отказа, если на этой конечной точке возникнет проблема или потребуется провести обслуживание.

С другой стороны, вложенные дочерние профили позволяют задать приоритет, согласно которому трафик всегда будет направляться на работоспособную конечную точку. Если конечная точка неработоспособна, трафик перенаправляется на альтернативную конечную точку. На рисунке 11.6 показано, что выполняется отработка отказа и трафик направляется в другой регион, хотя вы также могли создать несколько экземпляров веб-приложений в регионе «Запад США» и использовать метод взвешенной маршрутизации для дочернего профиля. Приступая к развертыванию среды своего приложения, подумайте о том, как лучше всего обеспечить высокую доступность конечных точек вне диспетчера трафика. В этих примерах настраивается отработка отказа между регионами, чтобы вы лучше могли оценить разницу в поведении.

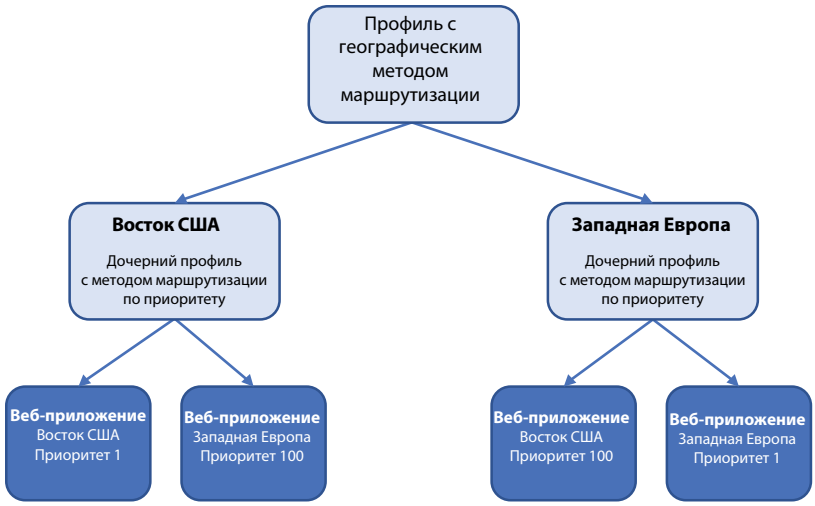


Рис. 11.6. Родительский профиль диспетчера трафика с методом географической маршрутизации должен использовать дочерние профили, которые содержат несколько конечных точек. Эти дочерние конечные точки могут использовать маршрутизацию по приоритету, чтобы всегда направлять трафик на предпочтительную конечную точку. Так, дочерний профиль в регионе «Восток США» всегда отправляет трафик на конечную точку в том же регионе при условии, что эта конечная точка нормально функционирует. Если конечная точка неисправна, трафик перенаправляется в Западную Европу. Без такого дочернего профиля клиенты на Востоке США не могли бы выполнить отработку отказа в альтернативную конечную точку и не получили бы доступ к вашему веб-приложению.

Попробуйте сейчас

Чтобы создать профили диспетчера трафика для своего распределенного приложения, выполните следующие действия.

В остальных упражнениях используются регионы «Восток США» и «Западная Европа». Если вы не живете в одном из этих регионов, выберите другой, наиболее подходящий для вас. Помните, что во всех упражнениях нужно использовать данные последовательно! В практическом упражнении в конце главы показано, как все это взаимодействует и работает, однако если вы живете за пределами Северной Америки или Европы и не измените регионы соответственно, система не сможет вас правильно перенаправить в ваши веб-приложения.

1. Откройте портал Azure и в верхней части панели щелкните значок Cloud Shell.
2. Создайте группу ресурсов, указав имя группы ресурсов, например `azuremolchapter11`, и расположение, например `eastus`:

```
az group create --name azuremolchapter11 --location eastus
```

3. Создайте родительский профиль диспетчера трафика. Нужно использовать метод географической маршрутизации и указать имя, например `azuremol`. Параметр имени DNS гласит, что оно должно быть уникальным, так что присвойте профилю уникальное имя. В следующем домене создается имя хоста azuremol.trafficmanager.net, которое используется для настройки веб-приложений в практическом упражнении в конце главы:

```
az network traffic-manager profile create \
  --resource-group azuremolchapter11 \
  --name azuremol \
  --routing-method geographic \
  --unique-dns-name azuremol
```

4. Создайте один из дочерних профилей диспетчера трафика. В этот раз используйте метод маршрутизации по приоритету и имя eastus, а затем укажите другое уникальное имя DNS, например `azuremoleastus`:

```
az network traffic-manager profile create \
  --resource-group azuremolchapter11 \
  --name eastus \
  --routing-method priority \
  --unique-dns-name azuremoleastus
```

5. Создайте еще один дочерний профиль диспетчера трафика с именем westeurope и еще одно уникальное имя DNS, например `azuremolwesteurope`:

```
az network traffic-manager profile create \
  --resource-group azuremolchapter11 \
  --name westeurope \
  --routing-method priority \
  --unique-dns-name azuremolwesteurope
```

6. Вы уже пару раз создавали веб-приложение, поэтому давайте используем интерфейс командной строки для быстрого создания 2 планов App Service, а затем создадим в каждом плане веб-приложение. Одно из этих веб-приложений расположено на Востоке США, а другое — в Западной Европе. В практическом упражнении в конце главы вы загружаете в эти веб-приложения примеры веб-страниц, поэтому пока просто создайте пустой веб-сайт и подготовьтесь к использованию локального репозитория Git.

   Чтобы создать веб-приложение на Востоке США, выполните следующие действия:

```
az appservice plan create \
  --resource-group azuremolchapter11 \
  --name appserviceeastus \
  --location eastus \
  --sku S1
az webapp create \
  --resource-group azuremolchapter11 \
  --name azuremoleastus \
  --plan appserviceeastus \
  --deployment-local-git
```

7. Создайте второе веб-приложение в Западной Европе:

```
az appservice plan create \
  --resource-group azuremolchapter11 \
```

```
    --name appservicewesteurope \
    --location westeurope \
    --sku S1
az webapp create \
    --resource-group azuremolchapter11 \
    --name azuremolwesteurope \
    --plan appservicewesteurope \
    --deployment-local-git
```

11.3.2 Глобальное распределение трафика до ближайшего экземпляра

Вы создали профили диспетчера трафика и конечные точки, но не сам трафик для перемещения. Если клиенты были направлены в профили, связи с вашими конечными точками не будет. Схема на рисунке 11.7 показывает, как связать конечные точки с профилями.

Рис. 11.7. В этом разделе вы свяжете свои конечные точки с профилями диспетчера трафика и определите приоритет распределяемого трафика.

Первые связи создаются для конечных точек вашего веб-приложения. Помните, что чтобы обеспечить высокую доступность, оба веб-приложения должны быть доступны каждому профилю диспетчера трафика. Используйте метод маршрутизации по приоритету для направления всего трафика в основное веб-приложение для каждого профиля. Если это веб-приложение недоступно, может быть выполнена отработка отказа, в результате чего трафик будет перенаправлен на конечную точку вторичного веб-приложения.

Когда в разделе 11.3.1 вы создавали профили диспетчера трафика, для параметров проверки работоспособности и мониторинга конечных точек было указано несколько значений по умолчанию. Рассмотрим эти варианты:

- *Время жизни (TTL) DNS: 30 с* — укажите, как долго можно кэшировать DNS-отклики от диспетчера трафика. Короткое время жизни гарантирует подобающую маршрутизацию клиентского трафика при внесении изменений в конфигурацию диспетчера трафика.

- *Протокол мониторинга конечной точки: HTTP* — можно также выбрать протокол HTTPS или наладить базовую проверку TCP. Как и в случае с балансировщиками нагрузки, протокол HTTP или HTTPS гарантирует возврат отклика HTTP 200 OK с каждой конечной точки.
- *Порт: 80* — порт для проверки на каждой конечной точке.
- *Путь: /* — по умолчанию проверяет корень конечной точки, хотя можно также настроить пользовательскую страницу (как, например, страница проверки работоспособности, используемая балансировщиками нагрузки).
- *Интервал зондирования конечной точки: 30 секунд* — как часто проверяется работоспособность конечной точки. Можно задать значение 10 или 30 секунд. При выполнении быстрого зондирования каждые 10 секунд за каждую конечную точку взимается дополнительная плата.
- *Допустимое количество сбоев: 3* — сколько раз конечная точка может не пройти проверку работоспособности, прежде чем конечная точка будет помечена как недоступная.
- *Тайм-аут зондирования: 10 с* — продолжительность периода до того, как зондирование будет помечено как неудачное и будет предпринята следующая попытка проверки конечной точки.

Менять эти значения по умолчанию не нужно. При создании собственных сред приложений для ключевых рабочих нагрузок в реальной жизни можно уменьшить число допустимых сбоев или интервал зондирования. Эти изменения гарантируют быстрое обнаружение любых проблем работоспособности и быструю маршрутизацию трафика на другую конечную точку.

Попробуйте сейчас

Чтобы связать конечные точки с профилями и завершить географическую маршрутизацию, выполните следующие действия:

1. На портале Azure найдите и выберите свою группу ресурсов. Для этого упражнения выберите профиль диспетчера трафика, созданный для Востока США.
2. На панели навигации в левой части профиля выберите конечные точки, а затем нажмите «Добавить».
3. Создайте конечную точку Azure и введите имя, например `eastus`.
4. Существуют различные типы целевых ресурсов. Вы будете использовать App Service. В качестве целевого ресурса выберите веб-приложение на Востоке США, например azuremoleastus.
5. Оставьте для параметра «Приоритет» значение 1, примите все другие значения по умолчанию и нажмите кнопку «ОК».
6. Повторите процесс, чтобы добавить другую конечную точку. На этот раз назовите конечную точку `westeurope`, выберите в качестве целевого ресурса веб-приложение в Западной Европе и задайте приоритет 100.

   Теперь в вашем профиле диспетчера трафика перечислены 2 конечные точки: одна — для веб-приложения на Востоке США, а другая — для веб-приложения в Западной Европе, как показано на рисунке 11.8. Маршрутизация конечных точек на основе приоритета всегда направляет трафик в веб-приложение на

Востоке США, если соответствующий ресурс работоспособен. Если ресурс недоступен, обеспеченная избыточность выполнит отработку отказа и перенаправит трафик в веб-приложение в Западной Европе.

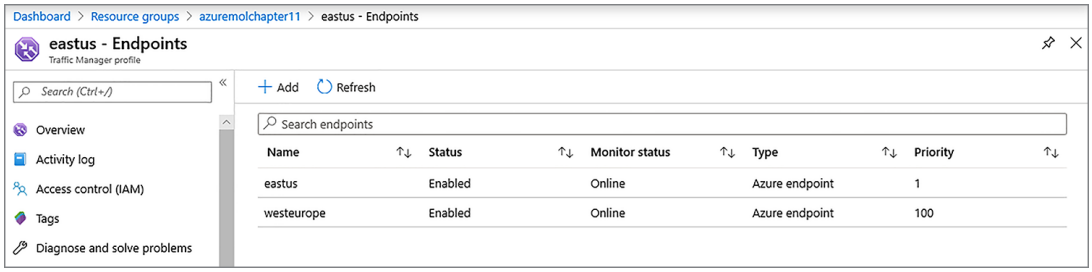


Рис. 11.8. Для профиля диспетчера трафика перечислены 2 конечные точки. Конечная точка для Востока США имеет низкое значение приоритета, так что она всегда принимает трафик, если работоспособна. Избыточность обеспечивается с помощью конечной точки в Западной Европе, которая используется только в том случае, если конечная точка на Востоке США недоступна.

7. Вернитесь в свою группу ресурсов и выберите профиль диспетчера трафика для Западной Европы.
8. Добавьте конечные точки.
9. Повторите процесс, чтобы добавить 2 конечные точки, и настройте их следующим образом:
   - Имя: westeurope
     Целевой ресурс: веб-приложение в Западной Европе
     Приоритет: 1
   - Имя: eastus
     Целевой ресурс: веб-приложение на Востоке США
     Приоритет: 100

Теперь в вашем профиле диспетчера трафика перечислены 2 конечные точки: одна — для веб-приложения на Востоке США, а другая — для веб-приложения в Западной Европе, как показано на рисунке 11.9. Вы обеспечили избыточность так же, как в предыдущем профиле диспетчера трафика, но в этот раз весь трафик отправляется в Западную Европу, если соответствующая конечная точка работоспособна, и на Восток США в противном случае.

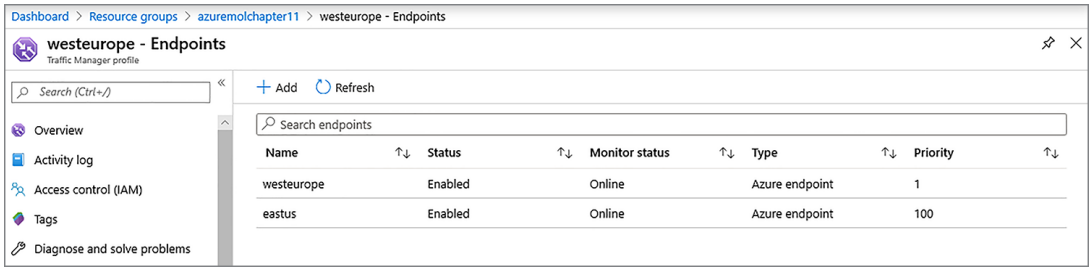


Рис. 11.9. Используется та же конфигурация конечных точек, как и в предыдущем профиле диспетчера трафика, однако в этот раз расположение веб-приложений зарезервировано. Дочерние профили можно использовать, чтобы всегда направлять в веб-приложения на Востоке США или в Западной Европе, однако сейчас вы обеспечили избыточность, позволяющую выполнять отработку отказа и переходить на другую конечную точку, если основная конечная точка в регионе недоступна.

И еще кое-что — последнее в этом разделе, я обещаю! Помните, что если вы используете диспетчер трафика для глобального распределения приложений, хорошей практикой считается обеспечение высокой доступности. В реальной жизни, возможно, ваша среда не столь сложна. Давайте посмотрим на схему еще раз и разберемся, какие дочерние профили и связи с региональными веб-приложениями требуется создать, как показано на рисунке 11.10.

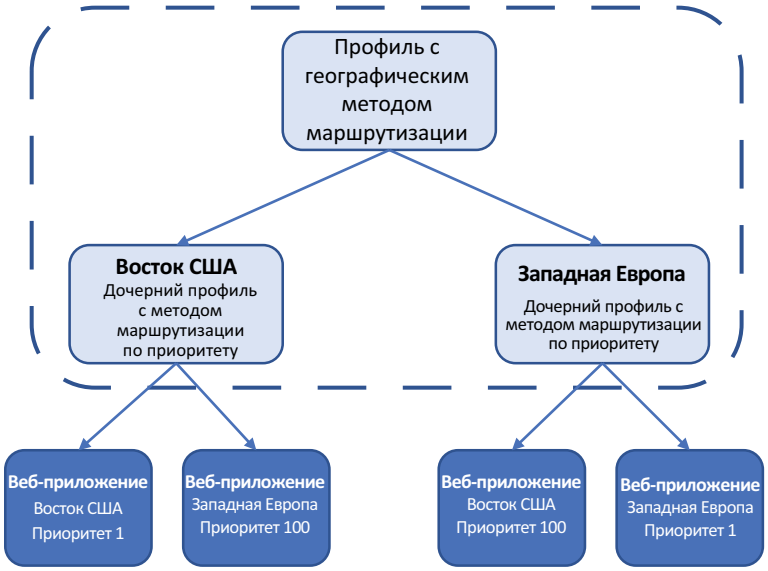


Рис. 11.10. Мы создали дочерние профили диспетчера трафика для Востока США и Западной Европы, а затем настроили должным образом региональные веб-приложения и значения приоритета. Теперь необходимо связать дочерние профили с родительскими.

Чтобы направлять трафик с учетом географического расположения, необходимо определить регион (например, Северную Америку) и вложенный профиль, например eastus. Все клиенты в регионе «Северная Америка» будут направляться в этот дочерний профиль. Вы настроили значения приоритетов для этого дочернего профиля, чтобы веб-приложение на Востоке США всегда обслуживало поступающий трафик. Однако вы обеспечили избыточность для отработки отказа и перевода при необходимости трафика на веб-приложение в Западной Европе.

Противоположное происходит для клиентов в Западной Европе. Можно добавить еще одну конечную точку для родительского профиля диспетчера трафика — на этот раз в Европе, потому что именно этот регион будет связан с конечной точкой, а затем добавить вложенный профиль `westeurope`. Весь европейский трафик направляется в этот профиль, а веб-приложение в Западной Европе всегда обслуживает это веб-приложение. Если возникает проблема, система выполняет отработку отказа и направляет трафик на Восток США.

Если существует политика или предписания о суверенитете данных, запрещающие перенаправление трафика в другой регион в случае отказа, возможно, потребуется скорректировать настройки конечных точек и профилей диспетчера трафика. Например, можно создать несколько веб-приложений в Западной Европе, как было показано

в главе 9. Так у вас будет несколько экземпляров веб-приложения, которые могут обслуживать клиентов. А если ваше приложение выполняется на виртуальных машинах, используйте набор масштабирования за пределами подсистемы балансировки нагрузки, чтобы обеспечить аналогичную избыточность профиля.

Попробуйте сейчас

Здесь важно ваше региональное расположение! Если вы находитесь за пределами региональных групп, показанных в профилях диспетчера трафика, обязательно выберите свой регион. В противном случае вы не сможете получить доступ к веб-приложению при выполнении практического упражнения в конце главы.

Чтобы связать дочерние профили с родительским профилем, выполните следующие действия:

1. На портале Azure найдите и выберите свою группу ресурсов.
2. Выберите родительский профиль диспетчера трафика. В предыдущих примерах ему было присвоено имя azuremol.
3. На панели навигации в левой части профиля выберите конечные точки, а затем нажмите «Добавить».
4. Создайте конечную точку, использующую первый дочерний профиль. В качестве типа выберите вложенную конечную точку и укажите имя, например `eastus`. В качестве целевого ресурса выберите профиль диспетчера трафика для Востока США.
5. В поле «Группировка по регионам» из раскрывающегося меню выберите «Северная Америка/Центральная Америка/Карибы» и нажмите кнопку «ОК».
6. Повторите шаги, чтобы добавить другую конечную точку. На этот раз назовите конечную точку westeurope, задайте в качестве целевого ресурса дочерний профиль диспетчера трафика для Западной Европы и выберите значение «Европа» из раскрывающегося меню в поле «Группировка по регионам».

   В ваших конечных точках для родительского профиля теперь указаны два дочерних профиля, а каждая конечная точка связана с соответствующим географическим регионом, как показано на рисунке 11.11.

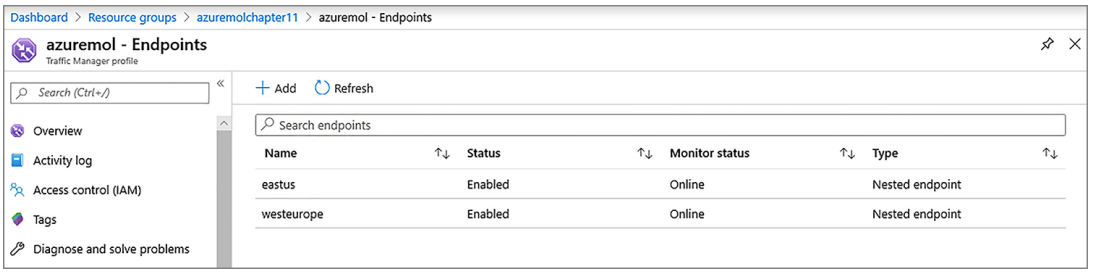


Рис. 11.11. Вложенные дочерние профили со связанными географическими регионами. Теперь этот родительский профиль диспетчера трафика направляет весь трафик из Европы в веб-приложение в Западной Европе, а при возникновении проблемы — на Восток США, поскольку в системе обеспечена избыточность. Противоположное верно для клиентов в Северной, Центральной Америке и странах Карибского бассейна.

В настоящее время веб-приложения настроены таким образом, что принимают трафик только в домене по умолчанию, который имеет вид *webappname*.azurewebsites.net. Когда диспетчер трафика направляет клиентов в эти экземпляры веб-приложения, трафик поступает из домена родительского профиля, например azuremol.trafficmanager.net. Веб-приложения не распознают этот домен, поэтому веб-приложение не загрузится.

7 Добавьте домен родительского профиля диспетчера трафика в оба экземпляра веб-приложения, созданных на шагах 4–6. При необходимости доменное имя можно получить на странице «Обзор» родительского профиля диспетчера трафика:

```
az webapp config hostname add \
  --resource-group azuremolchapter11 \
  --webapp-name azuremoleastus \
  --hostname azuremol.trafficmanager.net
az webapp config hostname add \
  --resource-group azuremolchapter11 \
  --webapp-name azuremolwesteurope \
  --hostname azuremol.trafficmanager.net
```

Теперь, когда вы переходите по адресу родительского профиля диспетчера трафика в веб-браузере, например https://azuremol.trafficmanager.net, you can't tell which endpoint you access, as both web apps run the same default web page. В практическом упражнении в конце главы вы загружаете основную веб-страницу в каждое веб-приложение, чтобы различать их!

Давайте проанализируем, что было создано в ходе этих упражнений. Это важно, потому что все функции высокой доступности и избыточности, описанные в предыдущих главах, теперь могут использоваться клиентами, а автоматическая маршрутизация трафика направляет их в ближайший экземпляр веб-приложения. В этой главе было создано следующее:

- Веб-приложение на Востоке США и еще одно — в Западной Европе.
- Профили диспетчера трафика, использующие географическую маршрутизацию для направления всех клиентов в Северной и Центральной Америке в веб-приложение на Востоке США, а всех клиентов из Европы — в веб-приложение в Западной Европе.
- Дочерние политики диспетчера трафика с маршрутизацией по приоритету, чтобы выполнять отработку отказа и в случае недоступности основного веб-приложения для региона перенаправлять трафик в другой регион.

С точки зрения высокой доступности:

- Если объединить эту настройку с автоматически масштабируемыми веб-приложениями, прямо сейчас можно обеспечить значительную избыточность.
- Если объединить эти веб-приложения с Cosmos DB, все ваше приложение будет автоматически масштабироваться, оно глобально распределено, а клиенты всегда осуществляют доступ к ближайшим к ним ресурсам, что гарантирует минимальную задержку, оптимальное время отклика и производительность.
- Даже если вы решили использовать виртуальные машины, для создания такой же высокодоступной глобально распределенной среды можно использовать масштабируемые наборы и балансировщики нагрузки.

И да, вы можете заменить диспетчер трафика на Front Door, если вам необходимы расширенные функции управления трафиком на уровне приложений.

Я знаю, что в последних главах было много нового для вас материала и изучение каждой из них заняло достаточно много времени на обеденном перерыве. Но только посмотрите, что вы узнали за последнюю неделю! Теперь вы можете создать веб-приложение с виртуальными машинами IaaS или веб-приложениями PaaS, обеспечить для них высокую доступность, балансировку нагрузки и возможность автоматического масштабирования, как показано на рисунке 11.12. Для своей базы данных можно использовать глобально распределенный сервер Cosmos DB и автоматически направлять клиентов в ближайший региональный экземпляр вашего приложения — и все это с помощью сервиса DNS, размещенного в Azure.

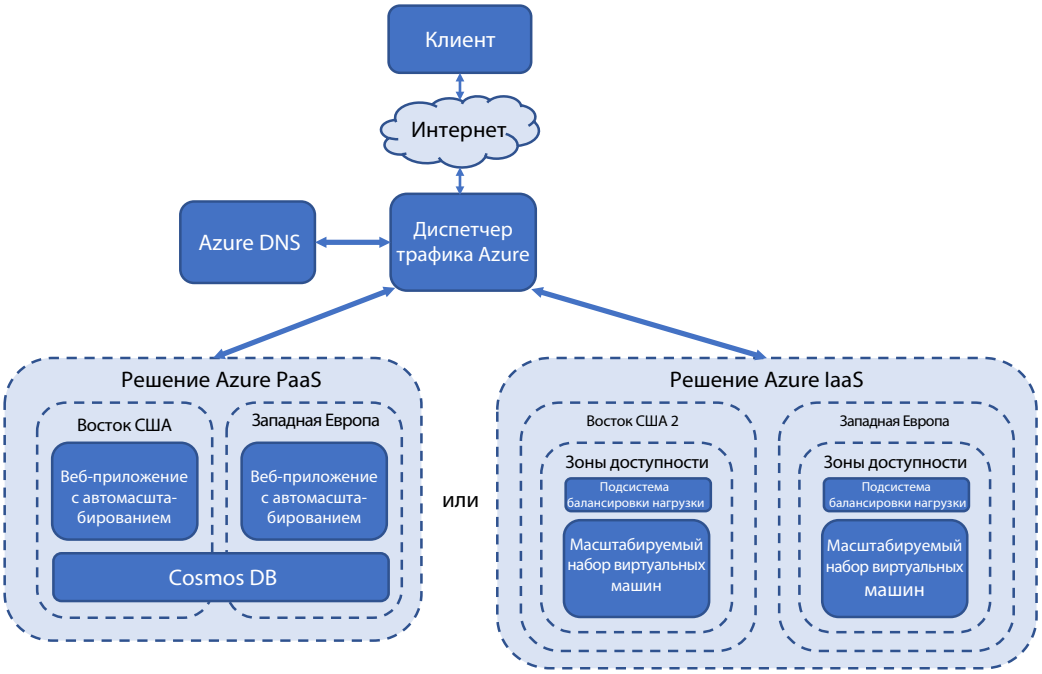


Рис. 11.12. Изучив последние несколько глав, вы научитесь создавать в Azure высокодоступные приложения IaaS или PaaS. В решениях IaaS можно использовать зоны доступности, балансировщики нагрузки и масштабируемые наборы. В решениях PaaS можно использовать веб-приложения с возможностью автоматического масштабирования и Cosmos DB. Диспетчер трафика и служба DNS Azure могут автоматически направлять клиентов в наиболее подходящий экземпляр приложения, учитывая их региональное расположение.

В практическом упражнении в конце главы пара основных веб-сайтов загружается в ваши веб-приложения, чтобы продемонстрировать, что диспетчер трафика работает и соответствующая конечная точка обслуживает ваш трафик. Если есть время, выполните упражнение; если нет — похлопайте себя по плечу и отправляйтесь вздремнуть. Я ничего не скажу боссу!

Во втором разделе книги есть еще одна глава, в которой рассматривается поддержание работоспособности приложений: как осуществлять мониторинг и диагностику приложений и инфраструктуры.

11.4 Практическое упражнение: развертывание веб-приложений для просмотра диспетчера трафика в действии

Из этой главы мы узнали много нового, и это упражнение поможет вам закрепить полученные знания и навыки использования Azure в работе с веб-приложениями. В примерах Azure репозиторий GitHub представляет собой две основные веб-страницы в онлайн-приложении магазина пиццы. Заголовок каждой веб-страницы указывает на расположение веб-приложения. Загрузите эти веб-страницы в соответствующий экземпляр веб-приложения, чтобы увидеть, как функционирует диспетчер трафика на практике:

1. При необходимости скопируйте репозиторий примеров GitHub в Cloud Shell следующим образом:

   ```
   git clone https://github.com/fouldsy/azure-mol-samples-2nd-ed.git
   ```

2. Начните с веб-страницы eastus, затем повторите следующие шаги в каталоге westeurope:

   ```
   cd ~/azure-mol-samples-2nd-ed/11/eastus
   ```

3. Инициализируйте репозиторий Git и добавьте основную веб-страницу:

   ```
   git init && git add . && git commit -m "Pizza"
   ```

4. На портале Azure в окне «Обзор» для вашего веб-приложения указан URL-адрес клона Git. Копируйте этот URL-адрес и задайте его в качестве целевого расположения вашего примера HTML-сайта в Cloud Shell с помощью указанной команды:

   ```
   git remote add eastus <your-git-clone-url>
   ```

5. Отправьте образец сайта в формате HTML в свое веб-приложение:

   ```
   git push eastus master
   ```

6. Повторите эти шаги в каталоге azure-mol-samples-2nd-ed/11/westeurope.
7. Затем откройте веб-браузер с доменным именем родительского профиля диспетчера трафика, например https://azuremol.trafficmanager.net, to see the traffic flow.

12 Мониторинг и устранение неполадок

В предыдущих главах вы узнали, как обеспечить высокую доступность своих приложений и направлять клиентов из разных точек земного шара в глобально распределенные экземпляры вашего приложения. Одна из целей заключалась в том, чтобы минимизировать объем взаимодействия с инфраструктурой приложения и предоставить платформе Azure возможность управлять работоспособностью и производительностью системы автоматически. Иногда все равно приходится засучить рукава и проверить результаты диагностики или показатели производительности. Из этой главы вы узнаете, как анализировать диагностику загрузки виртуальной машины, отслеживать показатели производительности и устранять проблемы подключения с помощью Наблюдателя сети.

12.1 Диагностика загрузки виртуальных машин

При работе с веб-приложениями вы развертываете код, а платформа Azure заботится обо всем остальном. В главе 3 мы рассмотрели основные аспекты диагностики и устранения проблем в развертываниях веб-приложений. Вы узнали, как анализировать события приложений в реальном времени, чтобы отслеживать их производительность. При работе с виртуальными машинами в облаке часто сложно диагностировать проблему, если нет возможности действительно увидеть экран компьютера и провести диагностику веб-приложения.

Одной из наиболее распространенных проблем с виртуальными машинами является отсутствие возможности подключения. Если невозможно подключиться к виртуальной машине по протоколу SSH или RDP, как выяснить, что не так? Во-первых, следует убедиться в том, что виртуальная машина работает должным образом. Для этого Azure предоставляет инструменты диагностики загрузки виртуальных машин, в том числе журналы загрузки и снимок экрана консоли.

Интерактивный доступ к консоли загрузки

В определенных сценариях диагностики и устранения неполадок можно также осуществлять доступ к интерактивной консоли последовательных подключений для ВМ в Azure. На консоли последовательных подключений можно выполнять интерактивный вход в систему и диагностировать проблемы загрузки. Можно перенастроить виртуальную машину так, чтобы исправлять все завершившиеся сбоем сценарии загрузки и неправильные конфигурации сервисов и приложений, мешающие правильной загрузке вашей виртуальной машины.

В этой главе не рассматриваются конкретные сценарии использования консоли последовательных подключений, однако это прекрасный ресурс, который в буквальном смысле поможет вам оказаться перед экраном запускающейся виртуальной машины. Кроме того, диагностика загрузки должна быть включена, поэтому эти упражнения обязательны для всех, кто планирует работать с консолью последовательных подключений.

Попробуйте сейчас

Чтобы создать виртуальную машину и включить диагностику загрузки, выполните следующие действия:

1. На портале Azure в левом верхнем углу выберите «Создать ресурс».
2. Найдите и выберите образ виртуальной машины Windows Server 2019 Datacenter.
3. Создайте группу ресурсов, например `azuremolchapter12`, а затем выберите ближайший к вам регион Azure.
4. Выберите размер ВМ, например DS1_v2.
5. Введите имя пользователя ВМ, например `azuremol`, и пароль. Пароль должен иметь длину не менее 12 символов и содержать любые 3 из следующих символов: строчная буква, прописная буква, цифра, специальный символ.
6. Примите любые параметры избыточности или правил входящих портов.
7. Примите значения по умолчанию для дисков и сети — вам ничего не нужно менять. Эти параметры должны быть вам уже знакомы.

   Возможно, ранее вы пропустили раздел, посвященный управлению. Как показано на рисунке 12.1, параметр «Диагностика загрузки» включен по умолчанию, создается учетная запись хранения.
8. Пока же оставим параметр «Диагностика гостевой ОС» отключенным.
9. Просмотрите параметры конфигурации ВМ и нажмите кнопку «Создать».

Для создания и настройки виртуальной машины требуется несколько минут, поэтому давайте продолжим изучать диагностику загрузки.

Если диагностика загрузки не включена и возникнет проблема, вы вряд ли сможете загрузить виртуальную машину, чтобы включить диагностику. Это как вечный спор о том, что было раньше, курица или яйцо, не так ли? Следовательно, диагностика загрузки автоматически включается для виртуальных машин, созданных на портале Azure. Необходимо включать диагностику загрузки вручную для Azure PowerShell, интерфейса командной строки Azure и наборов SDK для конкретных языков программирования. Я настоятельно рекомендую включать диагностику загрузки на виртуальных машинах в момент создания. Заведите привычку использовать шаблоны Azure Resource Manager (глава 6) либо собственные скрипты Azure CLI или PowerShell, которые включают диагностику загрузки во время развертывания.

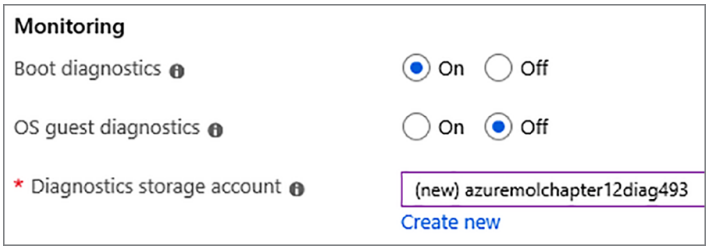


Рис. 12.1. По умолчанию диагностика загрузки включается при создании виртуальной машины на портале Azure. Создается учетная запись хранения, и именно в ней хранится диагностика загрузки. В одном из следующих упражнениях вы проанализируете и включите диагностику гостевой ОС, так что не нужно включать ее прямо сейчас). Для использования в производственной среде рекомендуется включить для каждой создаваемой ВМ и диагностику загрузки, и диагностику гостевой ОС.

Необходимо создать учетную запись хранения для журналов загрузки и снимков консоли, однако стоимость хранения этих данных в месяц вряд ли превысит 0,01 долл. США, если у вас нет очень интенсивной ВМ, которая генерирует очень много данных. Как только с вашими виртуальными машинами возникнет проблема и вам потребуется доступ к диагностике загрузки, вы поймете, что цент в месяц того стоит! Эту учетную запись хранения можно также использовать для хранения дополнительных метрик и журналов производительности на уровне ВМ, которые мы рассмотрим в разделе 12.2. Повторюсь, затраты на хранение таких данных минимальны. Даже по мере роста вашей среды виртуальных машин целесообразно понести небольшие дополнительные расходы, чтобы иметь возможность быстро диагностировать и устранить проблему, когда что-то идет не так.

Попробуйте сейчас

Чтобы просмотреть диагностику загрузки своей виртуальной машины, выполните следующие действия:

1. На портале Azure выберите в меню слева раздел «Виртуальные машины».
2. Выберите виртуальную машину, созданную в предыдущем упражнении.
3. В разделе «Поддержка и устранение неполадок» меню виртуальной машины выберите «Диагностика загрузки». Отобразятся диагностика загрузки и состояние виртуальной машины, как показано на рисунке 12.2. В отчете работоспособности содержится информация о том, возникали ли проблемы при загрузке виртуальной машины, и с его помощью, мы надеемся, вы сможете диагностировать проблему.

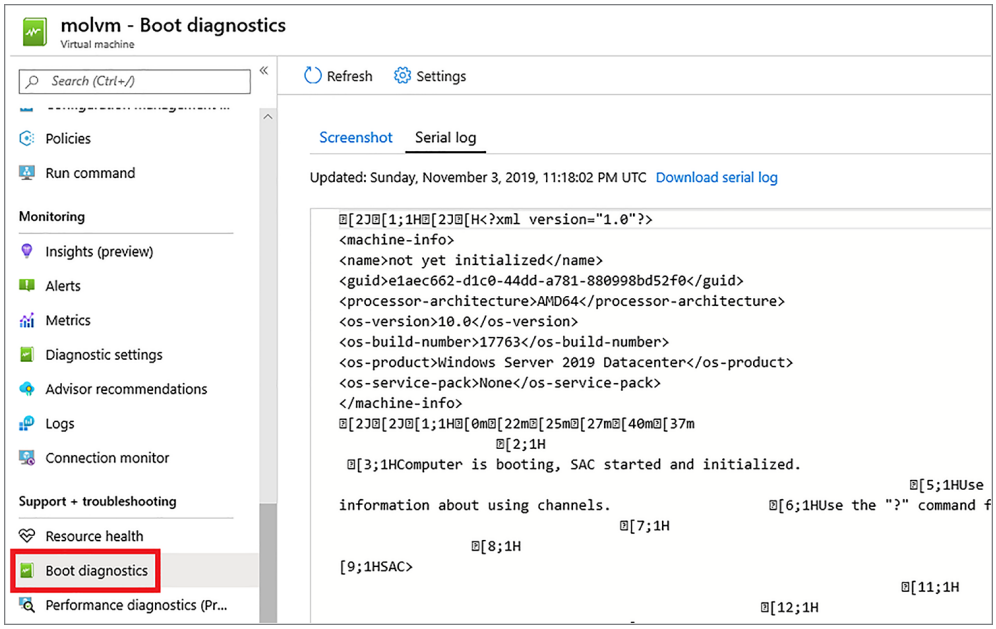


Рис. 12.2. Диагностика загрузки для отчета о работоспособности и состоянии загрузки виртуальной машины. Если отображаются ошибки, вы наверняка сможете диагностировать и устранить их причину. Кроме того, можно скачать журналы с портала и проанализировать их на локальном компьютере.

12.2 Метрики производительности и оповещения о ней

Одним из первых шагов в диагностике проблемы является анализ производительности. Сколько доступно памяти? Сколько потребляется ресурсов ЦП? Какова активность дисков?

При разработке и тестировании приложений в Azure я рекомендую фиксировать базовые показатели производительности в разные моменты. Эти базовые показатели дают представление о том, как должно работать ваше приложение под разной нагрузкой. Почему это важно? Как, скажем, через 3 месяца определить, что вы столкнулись с проблемами производительности, если отсутствуют данные, с которыми можно сравнить текущую производительность?

Когда в 9 главе вы научились автоматически масштабировать свои приложения, вы использовали базовые метрики производительности, например использование ресурсов ЦП, чтобы указать платформе Azure на необходимость увеличить или уменьшить число экземпляров вашего приложения. Эти базовые метрики дают лишь общее представление о том, как функционирует виртуальная машина. Для получения подробных метрик необходимо посмотреть на производительность ВМ. Для этого потребуется установить расширение диагностики Azure.

12.2.1 Просмотр метрик производительности с помощью расширения диагностики виртуальных машин

Чтобы расширить функциональность ваших виртуальных машин, в Azure предусмотрены десятки расширений, которые можно без проблем установить. Эти расширения устанавливают небольшой агент или среду выполнения приложения на виртуальную машину, чтобы возвращать диагностическую информацию на платформу Azure или в сторонние решения. Расширения ВМ могут автоматически настраивать и устанавливать компоненты или выполнять скрипты на ваших виртуальных машинах.

Расширение диагностики ВМ — это стандартное расширение, используемое для потоковой передачи метрик производительности с виртуальной машины в учетную запись хранения. Затем эти метрики производительности можно проанализировать на портале Azure или скачать и использовать в существующем решении мониторинга. Можно воспользоваться расширением диагностики, чтобы лучше разобраться в производительности ЦП и потреблении памяти внутри виртуальной машины. Как правило, это даст подробное и точное представление о происходящем, чем данные хоста.

Автоматизация и расширения ВМ

В главе 18 мы рассматриваем выполнение сервисом автоматизации Azure автоматизированных задач по расписанию на ваших виртуальных машинах. Одной из мощнейших возможностей сервиса автоматизации Azure является выполнение роли опрашивающего сервера настройки требуемого состояния (DSC) PowerShell. PowerShell DSC определяет заданное состояние желаемой настройки системы, устанавливаемых пакетов, файлов, разрешений и т. д. Вы создаете определения для требуемой настройки и применяете их к виртуальным машинам или физическим серверам. С помощью этих политик можно обеспечивать соблюдение требований и сообщать об этом. Расширение Azure PowerShell DSC используется для применения конфигурация DSC, например с опрашивающего сервера сервиса автоматизации Azure.

Среди других расширений, которые могут применять конфигурации и выполнять сценарии на виртуальных машинах, следует отметить настраиваемое расширение скриптов Azure. С помощью настраиваемого расширения скриптов вы либо определяете простой набор команд или указываете на один или несколько внешних сценариев, например размещенных в хранилище Azure или GitHub. Эти сценарии могут выполнять сложные задачи по настройке и установке. Кроме того, с их помощью можно гарантировать согласованную настройку всех развернутых виртуальных машин.

Расширение Azure PowerShell DSC и настраиваемое расширение скриптов обычно используются с масштабируемыми наборами виртуальных машин. Вы применяете одно из этих расширений к масштабируемому набору, а затем когда экземпляры ВМ создаются в масштабируемом наборе, они автоматически настраиваются для выполнения вашего приложения. Эти расширения служат для того, чтобы свести к минимуму неавтоматизированные операции по настройке ВМ, поскольку этот процесс сопряжен с большим количеством ошибок и требует вмешательства человека.

Другим способом автоматизации настройки виртуальных машин является использование сред Puppet и Chef — в обеих доступны расширения Azure для виртуальных машин. Если вы уже пользуетесь каким-либо инструментом управления конфигурацией, уточните у поставщика, какой подход он предпочитает использовать в Azure. Вероятно, существует расширение для виртуальных машин, которое сделает вашу жизнь значительно проще.

Попробуйте сейчас

Чтобы включить расширение диагностики виртуальных машин, выполните следующие действия:

1. На портале Azure выберите в меню слева раздел «Виртуальные машины».
2. Выберите виртуальную машину, созданную в предыдущем упражнении.
3. В разделе «Мониторинг» меню виртуальной машины выберите «Параметры диагностики».
4. Нажмите кнопку «Включить мониторинг на гостевом уровне».

Включение мониторинга на гостевом уровне занимает несколько минут. В это время Azure делает вот что:

- Устанавливает расширение диагностики виртуальной машины
- Настраивает расширение для потоковой передачи метрик гостевого уровня для следующих областей:
  - Логический диск
  - Память
  - Сетевой интерфейс
  - Процессы
  - Процессор
  - Система
  - Обеспечение возможности потоковой передачи журналов приложений, безопасности и системных журналов в хранилище Azure

Установив расширение диагностики, можно ограничить сбор данных, выбрав для включения в отчет только некоторые счетчики производительности. Вас можете интересовать только использование памяти или вы хотите включить сбор метрик Microsoft SQL Server. По умолчанию сбор метрик осуществляется каждые 60 секунд. Для своих приложений и инфраструктуры можно настроить любую периодичность выборки.

Расширение диагностики ВМ также может выполнять потоковую передачу файлов журналов с ВМ, позволяя централизовать журналы приложений, безопасности и системы для анализа или оповещения, как показано на рисунке12.3. По умолчанию фиксируются критические ошибки, ошибки и предупреждения, генерируемые журналами приложений и системными журналами, а также события безопасности для последующего аудита сбоев. Можно изменить фиксируемые уровни журнала и включить сбор данных журнала из IIS, журналов приложений и событий системы трассировки событий Windows (ETW). В процессе планирования и развертывания приложений необходимо определить, какие журналы необходимо собирать.

В виртуальных машинах с Windows нет ничего уникального. Аналогичным образом можно использовать расширение диагностики на виртуальных машинах с Linux, чтобы получить метрики производительности и наладить потоковую передачу разных журналов.

Если на вашей виртуальной машине возникнет проблема, единственным способом проанализировать произошедшее часто становится изучение *аварийных дампов*. Службы поддержки часто просят предоставить эти дампы, если требуется выявить основную причину проблемы. Как и в случае с диагностикой загрузки, не существует способа включить аварийные дампы «задним числом», чтобы узнать, почему что-либо пошло не так, поэтому определите, какие процессы нужно отслеживать, и настройте аварийные дампы заблаговременно. Например, можно осуществлять мониторинг процессов IIS и записывать полный аварийный дамп в хранилище Azure, если произойдет сбой процесса.

Вот еще несколько параметров, которые можно настроить для метрик гостевого уровня:

- *Приемники* позволяют настроить расширение диагностики ВМ для отправки определенных событий в Azure Application Insights. Application Insights обеспечивает полную прозрачность работы вашего кода.
- *Агент* позволяет задать квоту хранения всех ваших метрик (значение по умолчанию — 5 ГБ). Кроме того, можно включить сбор журналов для самого агента, а также удалить агент.

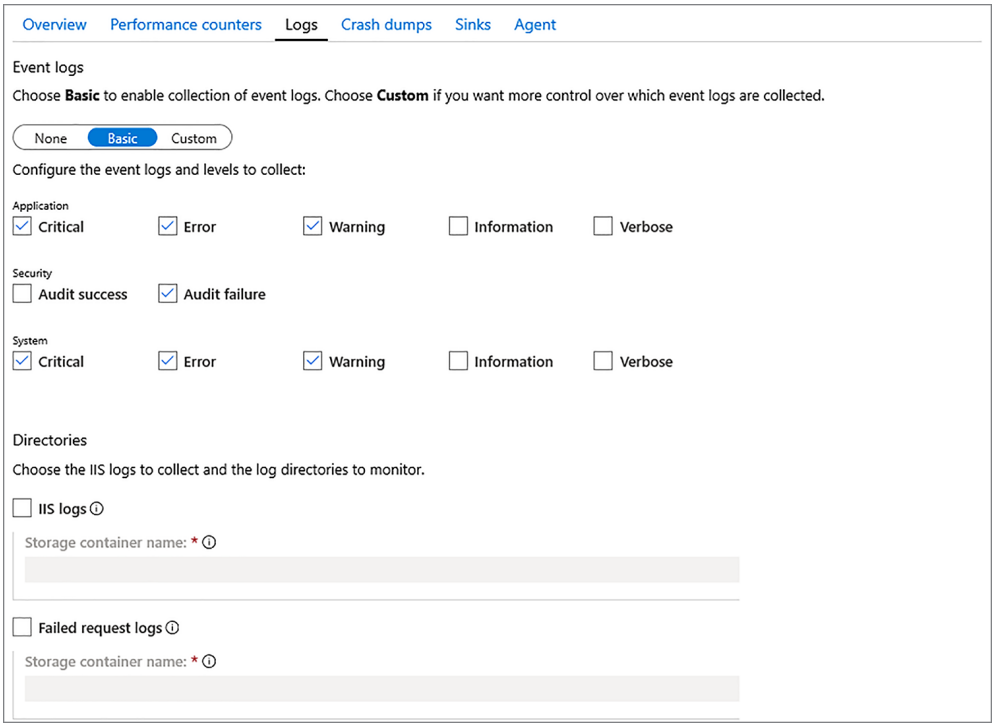


Рис. 12.3. На виртуальной машине можно настроить события и уровни журнала для различных компонентов. Эта возможность позволяет упорядочить журналы виртуальных машин для анализа и создания оповещений. Вам не нужно устанавливать сложные и, вероятно, дорогостоящие системы мониторинга — вы можете анализировать и получать уведомления, когда на ваших виртуальных машинах Azure возникают проблемы.

Попробуйте сейчас

Для просмотра метрик гостевого уровня выполните следующие действия:

1. На портале Azure выберите в меню слева раздел «Виртуальные машины».
2. Выберите виртуальную машину, созданную в предыдущем упражнении.
3. В разделе «Мониторинг» меню виртуальной машины выберите «Метрики».

Сейчас доступно гораздо больше метрик по сравнению с описанными в 9 главе метриками хостов. Изучите некоторые из доступных метрик хоста ВМ и гостевых систем и представьте приложения, для которых вам, возможно, потребуется отслеживать определенные метрики.

12.2.2 *Создание оповещений для условий производительности*

Как узнать, что возникла проблема, настроив свою виртуальную машину для отображения метрик производительности гостевого уровня? Конечно, никто не захочет сидеть, смотреть на графики производительности в реальном времени и ждать, пока что-то пойдет не так! Конечно, я не ваш босс, не мне вам указывать, но есть способ гораздо эффективнее : оповещения о метриках.

Оповещения о метриках позволяют выбрать ресурс, метрику и пороговое значение, а затем указать, кого и как следует уведомлять по достижении этого порогового значения. Оповещения работают не только для виртуальных машин. Так, можно определить оповещения для публичных IP-адресов, чтобы отслеживать входящие DDoS-пакеты и предупреждать вас о достижении определенного порога (что может свидетельствовать об атаке).

При генерировании оповещений можно настроить отправку уведомлений по электронной почте владельцам, участникам и читателям. Получить имена пользователей и их адреса электронной почты можно на основе действующей политики управления доступом на основе ролей. В крупных организациях подобная настройка может привести к рассылке уведомлений по электронной почте большому числу людей, так что используйте эту возможность с осторожностью. Кроме того, можно указать адреса электронной почты владельцев приложений или специалистов по проектированию архитектуры, составить список рассылки или группу получателей, объединяющую только заинтересованных лиц.

Существует еще несколько эффективных вариантов действий при получении оповещения:

- *Выполнение Runbook*. В главе 18 мы рассмотрим сервис автоматизации Azure. Он позволяет создавать и использовать модули Runbook, выполняющие скрипты. Эти сценарии могут выполнять базовые операции по восстановлению виртуальной машины, например перезапустить или даже перезагрузить виртуальную машину. Кроме того, они могут выполнять командлеты Azure PowerShell, чтобы, например, включить функции Наблюдателя сети Azure, такие как фиксация пакетов, которые подробнее рассматриваются далее в этой главе.
- *Запуск приложения логики.* Логические приложения Azure позволяют создавать рабочие процессы, выполняющие бессерверный код. Можно записать информацию в систему запросов в службу поддержки или инициировать автоматический телефонный вызов дежурному инженеру. В главе 21 мы исследуем удивительный мир бессерверных вычислений с использованием логических приложений Azure и функций Azure.

В практическом упражнении в конце главы вам нужно будет настроить несколько оповещений для виртуальной машины. Однако это не все возможности Azure по диагностике и мониторингу ваших виртуальных машин. Поговорим еще об одной распространенной причине неполадок — сети.

12.3 Наблюдатель за сетями Azure

Метрики производительности виртуальных машин и диагностика загрузки — это отличные способы мониторинга приложений Azure IaaS. Журналы веб-приложений и App Insights позволяют получить информацию о производительности ваших приложений PaaS. Сетевой трафик — это, конечно, не так гламурно, но часто именно он становится причиной проблем с подключением приложения, с которыми сталкиваетесь вы или ваши клиенты.

Еще в пятой главе я шутил о том, что специалистов по сетевым подключениям всегда винят во всем том, что не могут пояснить операционные специалисты. Однако мы можем помочь двум отделам наладить отношения или, по меньшей мере, получить неоспоримые доказательства, что виновата сеть! Наблюдатель сети Azure — это одна из функций, которая поможет нам восстановить отношения между специалистами. Наблюдатель сети позволяет осуществлять мониторинг и диагностику, используя следующие функции:

- Захват сетевых пакетов
- Проверка потока IP-адресов для групп безопасности сети
- Создание топологии сети

Эти функции прекрасны тем, что позволяют участвовать в диагностике и решении проблем разным специалистам. Если вы создали несколько виртуальных машин и не можете к ним подключиться, убедитесь в наличии сетевого подключения. Если созданное разработчиками приложение не может подключиться к серверному уровню базы данных, изучите правила групп безопасности сети и убедитесь, что проблема не кроется в этом. Специалисты по проектированию сетей могут захватывать пакеты и анализировать весь поток связи между хостами, тем самым проводя подробный анализ происходящего.

Дополнительные методы диагностики сети

Наблюдатель за сетями работает в тандеме с диагностическими журналами и метриками, описанными ранее в этой главе. Сетевые ресурсы, такие как балансировщики нагрузки и шлюзы приложений, могут также генерировать журналы диагностики. Эти журналы функционируют так же, как журналы приложений и системные журналы на виртуальной машине или в веб-приложении. Журналы сортируются на портале Azure, чтобы вам было легче определить наличие ошибок в конфигурации и связях между хостами и приложениями.

DNS и диспетчер трафика также имеют свою область диагностики на портале Azure. На портале представлена информация о наиболее распространенных ошибках, даны рекомендации по настройке и ссылки на дополнительную документацию. Если ничего не помогает, вы можете отправить запрос в службу поддержки Azure.

Несмотря на то что создавать и развернуть крупные приложения проще с помощью шаблонов диспетчера ресурсов Azure, интерфейса командной строки Azure или сценариев PowerShell, на портале Azure вы найдете массу полезных инструментов и функций для диагностики и устранения неполадок. Особенно если речь идет о сложных сетевых конфигурациях и политиках безопасности, потратьте пару минут на ознакомление с выходными данными инструментов Наблюдателя за сетями, и вы сможете выявить проблему и быстро решить ее. Все эти инструменты помогают повысить общую работоспособность и удобство эксплуатации ваших приложений для клиентов.

В каких случаях может потребоваться использовать Наблюдатель за сетями и его функции диагностики? Рассмотрим несколько распространенных проблем и как Наблюдатель за сетями может помочь нам в их решении.

12.3.1 Проверка потоков IP-адресов

Вот одна из распространенных проблем: клиенты не могут подключиться к приложению. Приложение работает нормально при подключении из офиса, но в общедоступном Интернете приложение недоступно для клиентов. Почему?

VPNs и ExpressRoute

Виртуальные частные сети (VPN) Azure обеспечивают безопасную связь между локальными офисами и центрами обработки данных Azure. Azure ExpressRoute предоставляет высокоскоростные, выделенные частные подключения между локальными офисами и центрами обработки данных Azure и часто используется в крупных организациях.

Оба подключения достаточно сложны, о них не расскажешь на обеденном перерыве. Кроме того, настраивают их обычно только один раз. Как правило, отвечают за это специалисты по сетевым подключениям, а пользователи часто даже не осознают, что осуществляют доступ к Azure по частному подключению.

Тестирование приложения проходит отлично. Приложение доступно в веб-браузере, вы можете размещать заказы и получать уведомления по электронной почте. Но когда разместить заказ пытаются ваши клиенты, приложение не загружается.

Как может помочь Наблюдатель за сетями? Проверяя потоки IP-адресов. Наблюдатель за сетями моделирует поток трафика в ваше целевое расположение и сообщает, достигает ли трафик вашей виртуальной машины.

Попробуйте сейчас

Чтобы включить Наблюдатель за сетями и проверить потоки IP-адресов, выполните следующие действия:

1. На портале Azure выберите «Все ресурсы» в верхнем меню навигации слева.
2. Отфильтруйте и выберите из списка доступных сервисов Наблюдатель за сетями. Включите Наблюдатель сети для регионов, в которых требуется осуществлять мониторинг. При включении Наблюдателя за сетями в определенном регионе Azure использует инструменты управления доступом на основе ролей для различных ресурсов и сетевого трафика.
3. Разверните список регионов для вашей учетной записи. Некоторые регионы уже могут быть включены. Если регион, в котором была развернута ВМ, не включен, выберите его, а затем включите Наблюдатель за сетями.
4. Если Наблюдатель за сетями включен в регионе (это занимает минуту или 2), выберите «Проверить поток IP-адресов» в разделе «Средства диагностики сети» в левой части окна «Наблюдатель за сетями».
5. Выберите группу ресурсов, например azuremolchapter12, и виртуальную машину, например molvm. По умолчанию задан протокол TCP, а направление — входящее. Также заполняется поле с локальным IP-адресом виртуального сетевого адаптера.
6. В поле «Локальный порт» укажите 80. Если при создании виртуальной машины в предыдущем упражнении вы приняли значения по умолчанию, порт 80 не был открыт, и сейчас у вас есть прекрасная возможность понаблюдать, что происходит в случае запрета трафика.
7. В разделе «Удаленный IP-адрес» введите `8.8.8.8`. Этот адрес может показаться вам знакомым — это открытый сервер DNS, предоставляемый Google. Вам не нужно что-либо делать с этим сервером — достаточно предоставить Наблюдателю сети внешний IP-адрес для моделирования потока трафика. Кроме того, можно перейти на сайт https://whatsmyip.com и ввести реальный публичный IP-адрес.
8. Задайте в поле «Удаленный порт» значение 80 и нажмите «Проверить».

Результатом проверки потока IP-адресов должно быть Отказано в доступе. К счастью, Наблюдатель за сетями показывает, какое правило привело к сбою потока трафика: DenyAllInBound. Вы знаете, что есть правило безопасности сети, которое блокирует трафик, но где применяется это правило? В подсети, на виртуальном сетевом адаптере или в группе безопасности приложений? Об этом вам расскажет Наблюдатель за сетями с помощью еще одной своей функции!

12.3.2 Просмотр действующих правил NSG

Правила NSG можно применять к отдельному виртуальному сетевому адаптеру, на уровне подсети, а также к группе виртуальных машин в группе безопасности приложений. Правила объединяются, благодаря чему можно задать общий набор правил для всей подсети, а затем детализировать эти правила для групп безопасности приложений (например «Разрешить TCP-порт 80 на всех веб-серверах») или отдельной ВМ.

Вот несколько примеров распространенных правил NSG, которые вы можете использовать:

- *Уровень подсети* — разрешить использовать TCP-порт 5986 для безопасного удаленного управления из подсети управления 10.1.10.20/24.
- *Уровень группы безопасности приложений* — разрешить использовать TCP-порт 80 для HTTP-трафика в веб-приложения и применять группу безопасности приложений ко всем ВМ веб-приложения.
- *Уровень виртуального сетевого адаптера* — разрешить использовать TCP-порт 3389 для удаленного доступа к рабочему столу из подсети управления 10.1.10.20/24.

Это базовые правила, которые явным образом разрешают определенный трафик. Если нет правил *Allow*, соответствующих сетевому пакету, по умолчанию применяются правила *DenyAll*, трафик не передается.

Во время тестирования приложения, рассмотренного в примере, вы, возможно, настроили HTTP-правило, разрешающее только трафик из одной из ваших локальных подсетей. Клиенты, пользующиеся публичным Интернетом, не могут подключиться к вашему приложению.

Попробуйте сейчас

Чтобы определить, где применяется правило NSG, выполните следующие действия:

1. В разделе «Наблюдатель за сетями» выберите «Действующие правила безопасности» слева.
2. Выберите группу ресурсов, например azuremolchapter12, и свою виртуальную машину, например molvm. Через несколько секунд отобразятся действующие правила, как показано на рисунке 12.4.

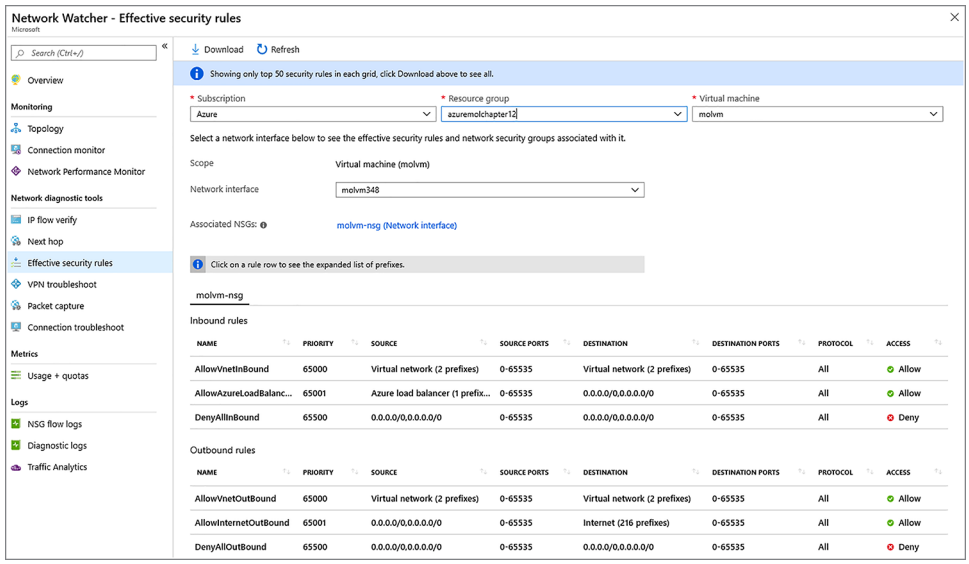


Рис. 12.4. Когда вы выбираете виртуальную машину, Наблюдатель за сетями изучает, как и в каком порядке применяются все правила NSG, а затем отображает информацию о действующих в настоящее время правилах. После этого вы можете быстро детализировать эти сведения до уровня подсети, виртуального сетевого адаптера и правил по умолчанию, чтобы найти и отредактировать место применения заданного правила.

Правила по умолчанию на виртуальной машине, которую вы создали ранее, вероятно, не самое интересное, что вы видели в жизни, однако можно быстро просмотреть правила подсети, сетевого интерфейса и правила по умолчанию, чтобы получить представление о том, как действуют правила в совокупности и как определить, где именно они применяются, если вам нужно внести изменения.

12.3.3 Захват сетевых пакетов

Допустим, вы обновили правила безопасности сети, чтобы предоставить клиентам доступ к вашему приложению из публичного Интернета, но один из клиентов жалуется, что система ведет себя странно. Иногда веб-приложение не загружается или отображает «битые» изображения. Часто происходит тайм-аут подключения.

Периодически возникающие проблемы диагностировать сложнее всего, особенно если ваш доступ к компьютеру, на котором наблюдается проблема, ограничен или отсутствует. Одним из распространенных методов диагностики является захват сетевых пакетов и их последующий анализ на наличие признаков проблем, таких как ошибки передачи по сети, неправильно сформированные пакеты, проблемы с протоколом и проблемы связи.

Захват сетевых пакетов позволяет вам получить доступ к необработанному потоку данных между двумя или более хостами. Анализ сетевых захватов — это настоящее искусство и уж точно занятие не для слабых духом! Специальные сторонние инструменты, например Riverbed Wireshark, Telerik Fiddler и Microsoft

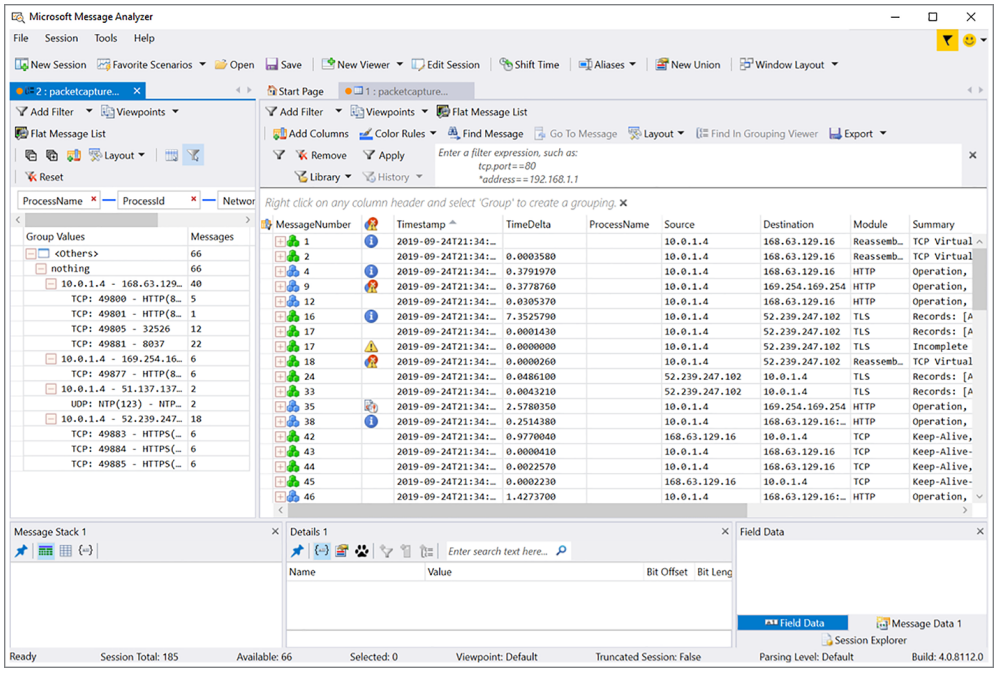


Рис. 12.5. Просмотр захвата сетевого пакета в Microsoft Message Analyzer. Можно проанализировать каждый пакет по отдельности. Можно группировать и фильтровать пакеты по протоколу связи или клиент-хосту. Столь глубокий анализ сетевых данных позволяет изучать фактические пакеты, которыми обмениваются узлы, с целью диагностики возникающих проблем. Бывший коллега однажды сказал мне: «Пакеты никогда не лгут». Самое сложное — понять, что «говорят» вам пакеты.

Message Analyzer, позволяют просматривать и фильтровать сетевые пакеты, которые обычно группируются по соответствующим связям или протоколам. На рисунке 12.5 показан пример захвата сетевых пакетов.

Чтобы включить захват пакетов, которые принимают и отправляют ваши виртуальные машины, Наблюдателем сети, сначала установите расширение виртуальной машины «Наблюдатель за сетями». Как вы видели в разделе 12.3.2, с помощью расширений ВМ платформа Azure может проникать внутрь ВМ и выполнять разные задачи управления. В случае с расширением «Наблюдатель сети» платформа анализирует сетевой трафик, который принимает и отправляет виртуальная машина.

Попробуйте сейчас

Чтобы установить расширение виртуальной машины «Наблюдатель сети» и приступить к захвату сетевых пакетов, выполните следующие действия:

1. На портале Azure в меню слева выберите «Виртуальные машины», а затем выберите свою виртуальную машину, например molvm.
2. В категории «Параметры» слева от окна виртуальной машины выберите «Расширения».
3. Нажмите кнопку «Добавить расширение».
4. В списке доступных расширений выберите «Агент Наблюдателя за сетями для Windows» и нажмите «Создать».
5. Чтобы подтвердить установку расширения, нажмите кнопку «ОК». Установка агента Наблюдателя за сетями на виртуальной машине может занять несколько минут.
6. Чтобы вернуться в меню Наблюдателя сети на портале Azure, выберите «Все ресурсы» вверху меню навигации «Службы» в левой части портала и нажмите «Наблюдатель за сетями».
7. В разделе «Инструменты диагностики сети» слева в окне Наблюдателя сети выберите «Захват пакетов» и нажмите «Добавить новый захват».
8. Выберите группу ресурсов, например azuremolchapter12, и ВМ, например molvm. Затем введите имя для захвата пакетов, например `molcapture`.

   По умолчанию захваты пакетов сохраняются в хранилище Azure. Можно также выбрать «Сохранить в файл» и указать локальный каталог на исходной виртуальной машине. Расширение агента Наблюдателя за сетями записывает файл захвата пакета на диск виртуальной машины.
9. Если еще не выбрано, выберите имя учетной записи хранения, которое начинается с имени вашей группы ресурсов, например azuremolchapter12diag493. Эта учетная запись хранения создается и используется расширением диагностики виртуальной машины, которое вы включили ранее.
10. Вы можете задать максимальный размер каждого пакета (по умолчанию — 0 для всего пакета), максимальный размер файла сеанса захвата пакетов (по умолчанию — 1 ГБ) и ограничение по времени для захвата пакетов (по умолчанию — 5 часов). Чтобы захватывать только трафик из определенных источников или портов, можно добавить фильтр и сузить область захвата пакетов.
11. Установите ограничение по времени, равное 60 секундам.
12. Чтобы начать захват пакетов, нажмите кнопку «ОК».

Захват начинается через пару минут. Когда выполняется захват, данные потоком передаются в учетную запись хранения Azure или локальный файл на виртуальной машине. Список захватов отображается на странице портала Наблюдателя за сетями. Если вы осуществляете потоковую передачу журналов в хранилище Azure, можно настроить отправку захвата непосредственно в учетную запись хранения и скачать файл захвата с расширением .CAP. Затем можно открыть захват пакета в программе анализа, как сказано в разделе 12.3.3. По сути, пример сетевого захвата, показанный ранее в этой главе на рисунке 12.5, и является захватом пакета Наблюдателем за сетями Azure!

12.4 Практическое упражнение: создание оповещений о производительности

Надеюсь, что рассмотрев диагностику виртуальных машин, метрики и функции Наблюдателя сети, вы получили некоторое представление о том, что Azure может помочь вам в диагностике проблем приложения. Некоторые возможности, такие как диагностика загрузки и расширение диагностики виртуальных машин, имеет смысл, только если вы включаете и настраиваете эти возможности во время развертывания виртуальных машин.

В этом практическом упражнении вы настроите несколько оповещений о метриках, чтобы узнать, о чем можно получать уведомления и как выглядят получаемые вами оповещения.

1. На портале Azure выберите созданную вами в предыдущих упражнениях виртуальную машину.
2. В разделе «Мониторинг» для виртуальной машины выберите «Оповещения».
3. Выберите «Создать правило оповещения», а затем добавьте условие, когда процент использования ЦП превышает в среднем 10 % за последние 5 минут. На диаграмме должны отображаться актуальные метрики, и вы можете скорректировать пороговое значение, инициирующее оповещение (в данном случае — 10 %).
4. Добавьте группу действий, а также задайте для нее имя и короткое имя. Для этого практического упражнения укажите для обоих параметров значение «azuremol». Группы действий позволяют определить повторно используемые наборы выполняемых шагов при появлении оповещений. Можно, например, отправить по электронной почте уведомление набору пользователей, запустить автоматизированный скрипт PowerShell или приложение логики Azure.
5. Ознакомьтесь с доступными типами действий, а затем выберите «Электронная почта/SMS/push-уведомление/голосовое уведомление».
6. Выберите способ получения уведомлений, например по электронной почте или в SMS-сообщении. Некоторые операторы могут взимать плату за SMS и голосовые уведомления.
7. После создания группы действий введите имя оповещения и укажите серьезность. Этот параметр полезен, если определено много предупреждений. Он помогает в сортировке и приоритизации первоочередных задач для выполнения.
8. Когда все будет готово, создайте правило. Для активации правила и создания уведомлений требуется от 10 до 15 минут.

Это базовый пример, поэтому представьте все существующие оповещения и уведомления для приложений и сервисов и подумайте, как вы можете использовать эту функцию при запуске рабочих нагрузок в Azure.

Часть 3

Безопасность по умолчанию

В интернет-мире, где приложения, как правило, подключены к Интернету в режиме 24/7, угроза цифровой атаки слишком очевидна. Эти атаки забирают ваше время, деньги и доверие клиентов. Одним из основных аспектов при разработке распределенных приложений с высоким уровнем избыточности является обеспечение их безопасности и защита ваших данных. В Azure предусмотрено несколько встроенных функций для защиты ваших данных, включая шифрование, мониторинг, цифровое хранилище ключей и резервное копирование. В этой части книги вы научитесь обеспечивать безопасность своих приложений с самого начала.

13 Резервное копирование, восстановление и репликация

В следующих нескольких главах представлены базовые функции и сервисы Azure для обеспечения безопасности ваших приложений. Это субъективное мнение, но я считаю, что безопасность не должна являться какой-то дополнительной функцией или предметом для размышлений. Безопасность необходимо интегрировать в ваше приложение уже на ранних этапах разработки. В этой главе мы начнем рассмотрение вопросов безопасности Azure с рассмотрения способов резервного копирования и восстановления ваших данных. Вам может показаться, что резервное копирование слабо связано с безопасностью, однако не ограничивайте вопросы безопасности шифрованием данных или SSL-сертификатами веб-сайтов. Как же защитить данные в случае отключения систем, утери данных и хакерских атак? Кроме того, обсуждение резервного копирования и репликации позволит плавно перейти от предыдущей главы посвященной высокой доступности, к этой главе.

Резервное копирование может показаться чем-то обыденным, и как администратор по резервному копированию в прошлом могу подтвердить, что создание и ротация резервных копий — не самое увлекательное занятие в мире! Однако своевременное создание резервных копий, которые действительно работают, помогут защитить ваши приложения и — в самом худшем случае — быстро и надежно восстановить ваши данные. Кроме того, можно реплицировать виртуальные машины из одного региона Azure в другой. Эта возможность обеспечивается высокой доступностью, которую мы рассматривали в главе 7.

Из этой главы вы узнаете, как создавать резервные копии виртуальных машин, восстанавливать и автоматически реплицировать виртуальные машины в Azure. На всех этапах резервного копирования и восстановления в целях безопасности данные шифруются.

13.1 Сервис архивации Azure

Одно из ключевых преимуществ сервиса архивации Azure — это то, что это одновременно и сервис, и большой ресурс для хранения реальных резервных копий. Сервис архивации

Azure может защитить виртуальные машины в Azure, локальные ВМ, физические серверы и даже ВМ других поставщиков, например Amazon Web Services (AWS). Резервные копии данных можно сохранить в собственных локальных массивах хранения или в хранилище восстановления Azure. На рисунке 13.1 показано, как сервис архивации Azure может обеспечить безопасность ваших данных и удовлетворить все потребности в сфере резервного копирования.

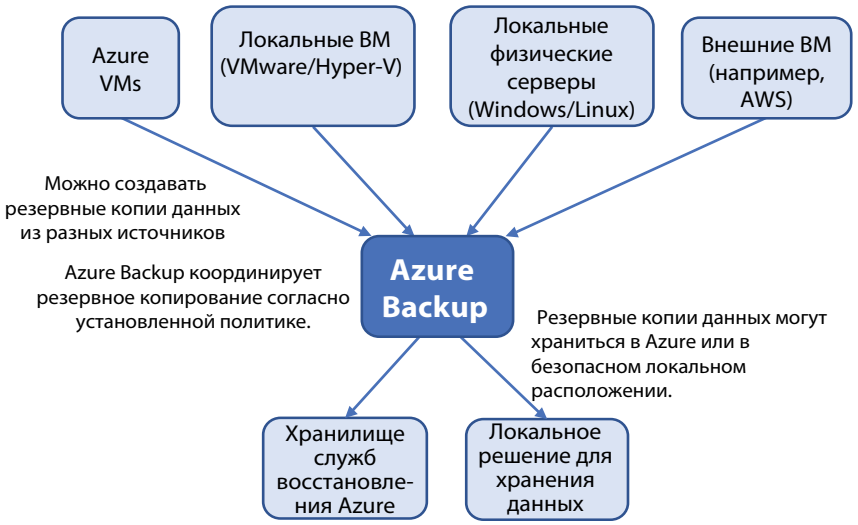


Рис. 13.1. Центральный сервис оркестрации позволяет создать резервные копии нескольких виртуальных машин или физических серверов (разных поставщиков и из разных расположений). Сервис архивации Azure использует определенные политики для резервного копирования данных с определенной периодичностью или по расписанию. Затем эти резервные копии можно хранить в Azure или в локальном хранилище. Кроме того, на протяжении всего процесса данные зашифрованы, что обеспечивает дополнительную безопасность.

По сути, сервис архивации Azure управляет графиком резервного копирования и хранением данных, а также координирует задания резервного копирования или восстановления данных. Для резервного копирования виртуальных машин Azure не требуется вручную устанавливать какие-либо серверные компоненты или агенты. Все операции по резервному копированию и восстановлению интегрированы в платформу Azure.

Для резервного копирования локальных виртуальных машин, физических серверов или виртуальных машин других поставщиков (например, AWS) необходимо установить небольшой агент, который обеспечит безопасный обмен данными с Azure. Подобный безопасный обмен данными гарантирует шифрование ваших данных во время передачи.

Что касается данных, хранимых в Azure, резервные копии шифруются с помощью ключа шифрования, созданного вами и доступного только вам. Доступ к зашифрованным резервным копиям есть только у вас. Кроме того, можно создать резервную копию зашифрованных виртуальных машин (мы поговорим об этом в главе 14) — так резервные копии ваших данных будут действительно надежно защищены.

Резервное копирование и восстановление данных не требует изменения потока сетевого трафика. Вы платите только за каждый защищенный экземпляр, а затем за реально потребленные ресурсы хранения в Azure. Если вы пользуетесь локальным хранилищем, стоимость использования сервиса архивации Azure минимальна, поскольку затраты на хранилище Azure или передачу трафика отсутствуют.

13.1.1 Политики и хранение

Сервис архивации Azure использует модель добавочного резервного копирования. Если вам нужно обеспечить безопасность экземпляра, первая операция резервного копирования выполняет полное резервное копирование данных. После этого каждая операция резервного копирования выполняет добавочное резервное копирование данных. Каждая из этих резервных копий называется точкой *восстановления*. Добавочное резервное копирование — эффективный с точки зрения экономии времени подход, оптимизирующий использование ресурсов хранения и сетевой полосы пропускания. Только данные, которые изменились с момента предыдущего резервного копирования, безопасно передаются в целевое расположение резервного копирования. На рисунке 13.2 представлена подробная схема добавочного резервного копирования. С помощью сервиса архивации Azure можно

Рис. 13.2. При использовании добавочного резервного копирования копируются только данные, которые изменились с момента выполнения предыдущей операции. При первом резервном копировании всегда создается полная резервная копия. При последующих заданиях резервного копирования копируются только данные, изменившиеся с момента выполнения предыдущего задания. Периодичность создания полных резервных копий регулируется с помощью политик. Такой подход позволяет свести к минимуму объем данных, которые необходимо безопасно передать по сети и разместить в целевом расположении хранения. Сервис архивации Azure поддерживает связь между добавочными резервными копиями, так что вы восстанавливаете согласованные и полные данные.

сохранить до 9999 точек восстановления для каждого экземпляра, безопасность которого вы обеспечиваете. Так, если вы будете выполнять резервное копирование ежедневно, этого ресурса хватит как минимум на 27 лет. Если создавать резервные копии еженедельно, ресурса хватит почти на 200 лет. Думаю, это удовлетворит большинство аудиторов! Можно хранить ежедневные, еженедельные, ежемесячные и ежегодные резервные копии (это соответствует большинству существующих политик резервного копирования).

Чтобы реализовать оптимальную стратегию резервного копирования для своей рабочей нагрузки, необходимо понять и определить приемлемые *целевую точку восстановления* (RPO) и *целевое время восстановления* (RTO).

Целевая точка восстановления

RPO определяет точку, которую можно восстановить из самой свежей резервной копии. По умолчанию сервис архивации Azure создает резервные копии ежедневно. Затем необходимо сформулировать политику хранения (сколько дней, недель, месяцев или лет) следует хранить эти точки восстановления. Несмотря на то что RPO обычно служит для определения максимального допустимого объема потери данных, нужно учитывать и то, насколько далеко назад во времени вы захотите переместиться, если что-то произойдет. На рисунке 13.3 показано, как сервис определяет допустимый объем потери данных.

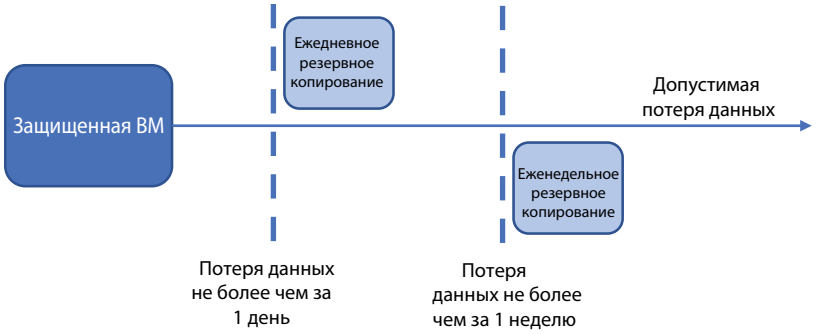


Рис. 13.3. Целевая точка восстановления (RPO) определяет, какой объем данных защищаемого экземпляра будет для вас приемлемым. Чем больше RPO, тем больше допустимая потеря данных. RPO длительностью один день означает, что потеря данных за период до 24 ч (в зависимости от того, когда после последнего резервного копирования произошла потеря данных) является для вас приемлемой. RPO длительностью одну неделю означает, что потеря данных за неделю является приемлемой.

Крупные аварии и потери больших объемов данных случаются редко. Гораздо чаще случаются незначительные потери данных или перезапись. Зачастую такие инциденты замечают и сообщают о них только спустя некоторое время после потери данных. И тут на первый план выходит политика хранения, настроенная для ваших защищаемых экземпляров. Если период хранения данных согласно политике небольшой, возможно, вам не удастся восстановить их за нужный момент. Нужно равновесие между хранением большого числа точек восстановления и стоимостью хранения всех этих точек восстановления.

Хранилище Azure относительно экономично: как правило, менее 0,02 долл. США за 1 ГБ хранилища. Это примерно 2 долл. США в месяц за резервную копию данных ВМ объемом 100 ГБ (плюс плата за фактический сервис архивации Azure). В зависимости от того, насколько меняются ваши данные, число (и размер) добавочных точек восстановления может быстро увеличиваться. Если вы будете хранить точки восстановления неделями или месяцами, это может обойтись вам в десятки долларов в месяц за защищенный экземпляр. Я не пытаюсь вас отговорить, но важно планировать свои потребности и эффективно управлять расходами. Хранение данных выглядит дешево (всего 0,02 USD за ГБ), пока вы не накопите сотни ГБ данных на один защищенный экземпляр и десятки или даже сотни экземпляров, безопасность которых вам нужно будет обеспечивать.

Как бывший администратор по резервному копированию могу сказать, что при определении числа сохраняемых точек восстановления ключевым фактором часто становилась емкость хранилища. Именно емкость хранилища часто позволяла прийти к компромиссному решению по поводу RPO. Если вы используете хранилище Azure, а не локальное хранилище, вам не нужно беспокоиться о доступном объеме. Могу гарантировать: ресурсы Azure ограничены исключительно вашей платежеспособностью!

Целевое время восстановления

Целевое время восстановления определяет, насколько быстро вы можете восстановить свои данные. Если вы решили создать резервную копию ВМ Azure и сохранить точки восстановления в локальном хранилище, на восстановление резервных копий потребуется гораздо больше, чем если бы они размещались непосредственно в хранилище Azure. Противоположное верно, если вы создали резервную копию локальных ВМ или физических серверов и сохранили их в хранилище Azure. На рисунке 13.4 изображено RTO.

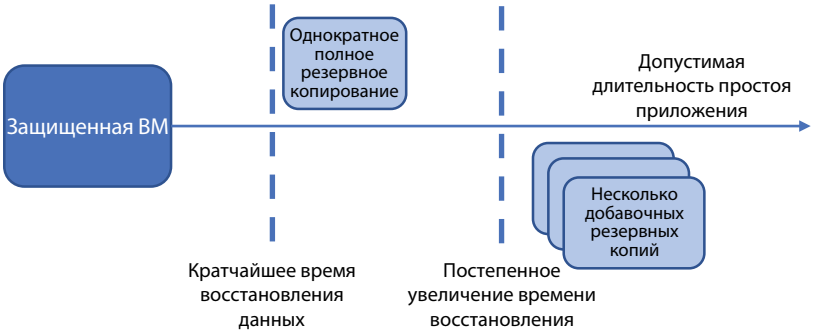


Рис. 13.4. Целевое время восстановления определяет допустимую продолжительность процесса восстановления данных (это время, когда приложение будет недоступно). Чем больше точек восстановления задействовано в процессе восстановления, тем больше целевое время восстановления. С другой стороны, чем ближе хранилище резервных копий к точке восстановления, тем меньше целевое время восстановления.

В любом из сценариев данные точки восстановления потребуется передать из расположения хранения точки восстановления в расположение восстановления. В крупных операциях восстановления, где может потребоваться передать сотни ГБ, настоящим «узким местом» становится пропускная способность сети: именно она определяет, как быстро приложения снова станут доступны.

То же самое происходит, если вы настроили длительное хранение с множеством последующих добавочных точек восстановления. Восстановление данных может потребовать установки и восстановления большого числа точек восстановления. Ваша задача — определить, насколько далеко назад вам потребуется переместиться во времени и сколько времени вы можете потратить на восстановление данных.

Значения RPO и RTO варьируются в зависимости от ваших приложений и использовании в бизнесе. Длительные отключения и простои недопустимы для приложения, которое обрабатывает заказы в реальном времени, поэтому показатели RPO и RTO наверняка будут очень низкими. Как правило, база данных используется для хранения данных, поэтому обычно вы укажете некие допуски в приложении, вместо того чтобы использовать точки восстановления. Если вспомнить базу данных Cosmos DB, станет ясно, что создавать резервную копию не из чего — платформа Azure выполняет репликацию и защиту данных за вас. При создании пользовательского решения в MySQL или Microsoft SQL Server, как правило, используется схожий тип кластеризации и репликации — это позволяет убедиться в наличии множества копий базы данных (так что потеря одного из экземпляров не потребует восстановления из резервной копии). Резервные копии используются, в основном, для защиты в случае крупных отключений или серьезного повреждения данных.

13.1.2 Расписания резервного копирования

Как контролировать периодичность создания резервных копий и время хранения точек восстановления? в службе архивации Azure эти параметры определяются с помощью политик. Вы формулируете эти политики, чтобы охватить разнообразные сценарии, от которых требуется защитить систему. Одни и те же политики можно использовать в работе с несколькими защищенными экземплярами.

Допустим, с помощью политики резервного копирования можно настроить создание резервной копии ежедневно в 18:30. Требуется хранить ежедневные резервные копии в течение 6 месяцев, меняя их таким образом, чтобы еженедельные резервные копии хранились в течение 2 лет. Для обеспечения соответствия требованиям вы храните ежемесячные резервные копии 5 лет. Ежегодная резервная копия хранится 10 лет. Продолжительность хранения может показаться чрезмерной, однако к приложениям, которые осуществляют обмен данными и приложениями, часто предъявляются столь жесткие законодательные и нормативные требования в отношении хранения резервных копий. Сервис архивации Azure позволяет свободно определять политики, соответствующие разным рабочим нагрузкам приложений, и быстро обеспечивать соответствие нормативным требованиям.

Попробуйте сейчас

Все ваши резервные копии Azure хранятся в хранилище служб восстановления. Чтобы создать хранилище и политику резервного копирования, выполните следующие действия:

1. Откройте портал Azure и в верхнем левом углу меню выберите «Создать ресурс».
2. Найдите и выберите «Backup and Site Recovery», а затем нажмите кнопку «Создать».
3. Создайте группу ресурсов, например `azuremolchapter13`, и введите имя хранилища, такое как `azuremol`.
4. Выберите расположение, проверьте данные и создайте хранилище.
5. После создания хранилища выберите «Группы ресурсов» в меню в левой части портала, а затем выберите созданную группу ресурсов.
6. Выберите хранилище сервисов восстановления из списка доступных ресурсов, выберите «Политики резервного копирования» в меню слева и нажмите «Добавить политику».
7. Выберите тип политики «Виртуальная машина Azure» и укажите имя своей новой политики, например `molpolicy`. По умолчанию резервная копия создается каждый день.
8. Выберите из раскрывающегося меню наиболее подходящий часовой пояс. По умолчанию в Azure используется всемирное координированное время (UTC).

   При желании можно скорректировать политики хранения ежедневных, еженедельных, ежемесячных и ежегодных резервных копий. В разделе, посвященном расписанию резервного копирования и хранения, подробно описано, как можно выбрать эти значения. Как правило, они изменяются при создании и применении политик резервного копирования для защиты экземпляров ВМ.
9. Когда все значения будут заданы, выберите «Создать».

Нет ничего проще

Настроить резервное копирование ВМ можно при создании ВМ на портале Azure. На странице Settings (Настройки), где настраиваются параметры виртуальной сети, диагностики и устранения неполадок, можно включить службу резервного копирования Azure. Можно выбрать существующее хранилище сервисов восстановления или создать новое, а также создать новую политику резервного копирования или использовать существующую. В настоящее время невозможно включить создание резервных копий в рамках развертывания ВМ в интерфейсе командной строки Azure или Azure PowerShell, однако после развертывания для этого обычно достаточно одной команды.

Мне нравится тщательно планировать стратегию резервного копирования, политики хранения и расписания, поэтому в упражнениях мы сначала создали хранилище сервисов восстановления и соответствующие политики. Однако если вам нужно быстро создать ВМ и включить резервное копирование, все это можно сделать на портале Azure за один шаг.

Теперь у вас есть политика резервного копирования, которая также определяет политики хранения в течение определенных периодов, однако пока вам нечего копировать. Давайте создадим виртуальную машину с Cloud Shell, чтобы вы могли создать резервную копию и в дальнейшем реплицировать данные.

Попробуйте сейчас

Чтобы создать тестовую виртуальную машину для резервного копирования и репликации, выполните следующие действия:

1. Щелкните значок Cloud Shell в верхней части портала Azure.
2. Создайте ВМ с помощью команды `az vm create`. Введите имя группы ресурсов, созданной в предыдущем практическом упражнении, например `azuremolchapter13`, и укажите имя ВМ, например `molvm`:

```
az vm create \
  --resource-group azuremolchapter13 \
  --name molvm \
  --image win2019datacenter \
  --admin-username azuremol \
  --admin-password P@ssw0rdMoL123
```

Политика резервного копирования определена, тестовая ВМ готова. Чтобы увидеть сервис архивации Azure в действии, давайте применим политику резервного копирования к виртуальной машине.

Попробуйте сейчас

Чтобы создать резервную копию виртуальной машины, используя определенную политику, выполните следующие действия:

1. На портале в меню слева выберите «Группы ресурсов».
2. Выберите группу ресурсов и созданную вами ВМ.
3. В разделе «Операции» выберите «Резервное копирование».
4. Убедитесь, что выбрано хранилище сервисов восстановления, и выберите политику резервного копирования из раскрывающегося меню.

5 Изучите параметры расписания резервного копирования и хранения, а затем включите резервное копирование. Применение политики резервного копирования занимает несколько секунд.

6 После включения политики вернитесь к параметрам резервного копирования. ВМ будет в состоянии Предупреждение (ожидание начальной архивации).

7 Чтобы создать первую резервную копию, нажмите кнопку «Создать резервную копию», как показано на рисунке 13.5.

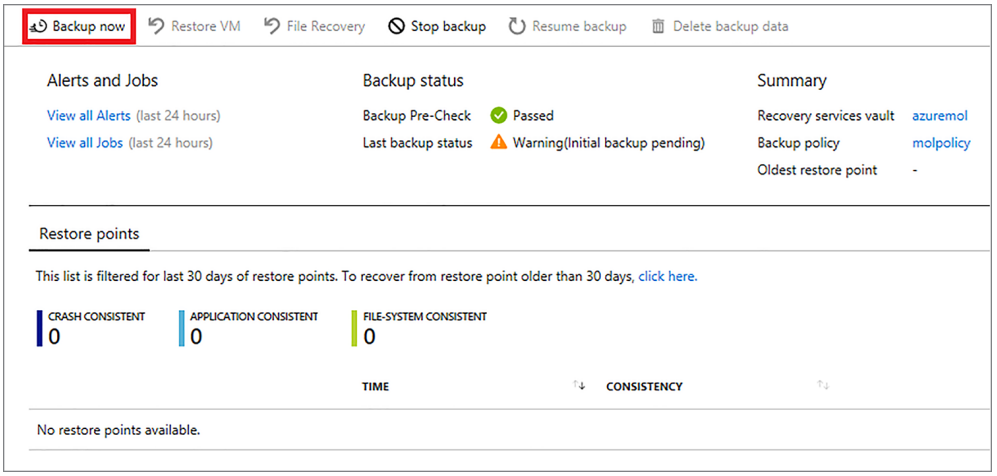


Рис. 13.5. Чтобы создать первую резервную копию, нажмите кнопку «Создать резервную копию». По окончании процесса статус обновляется и показывает время последнего резервного копирования, последнюю и самую старую точки восстановления.

Первое полное резервное копирование может занимать 15–20 минут. Чтобы отследить ход резервного копирования, можно выбрать «Просмотреть все задания». Индикатор хода выполнения или индикатор с процентами отсутствует, но вы можете убедиться, что задание все еще выполняется.

Это все, что нужно для резервного копирования виртуальных машин и защиты ваших данных в Azure! Далее я расскажу о том, как восстановить данные, если что-то пойдет не так.

13.1.3 Восстановление виртуальной машины

Сервис архивации Azure позволяет выполнить полное восстановление виртуальной машины или восстановление на уровне файлов. Многолетняя практика показывает, что чаще всего используется восстановление на уровне файлов. Как правило, этот тип восстановления выполняется, когда файлы удаляются или случайно перезаписываются. Восстановление на уровне файлов, как правило, определяет политики хранения ваших резервных копий. Чем важнее данные, тем выше вероятность, что вы захотите сохранить их резервные копии на длительный срок (на случай, если среди ночи вам позвонят и попросят восстановить сохраненный полгода назад файл).

Как легко предположить, при полном восстановлении виртуальной машины восстанавливается вся виртуальная машина. Мне редко приходилось выполнять полное восстановление виртуальной машины, чтобы удаленная ВМ снова оказалась в сети. Отличным

вариантом использования полного восстановления ВМ является создание тестовой виртуальной машины, которая была бы функционально эквивалентна оригиналу. Можно восстановить виртуальную машину, а затем протестировать на ней обновление программного обеспечения или любую другую процедуру обслуживания. Это может помочь выявить потенциальные проблемы и создать план действий для реальной производственной ВМ.

Очень важно регулярно тестировать свои резервные копии. Не ждите, пока в реальной жизни возникнет необходимость восстановить данные. Сервису архивации Azure можно доверять, но необходимо убедиться, что вы знаете, как и куда при необходимости восстановить данные!

Восстановление на уровне файлов

Восстановление на уровне файлов — отличная функция сервиса архивации Azure. Azure создает сценарий восстановления, который вы скачиваете и запускаете, что обеспечивает свободу выбора способа и места восстановления файлов. Сценарий восстановления защищен паролем, так что выполнить восстановление можете только вы. При выполнении скрипта восстановления отображается запрос на ввод пароля. Продолжить можно только после успешного ввода пароля. Окно скачивания сценария восстановления показано на рисунке 13.6.

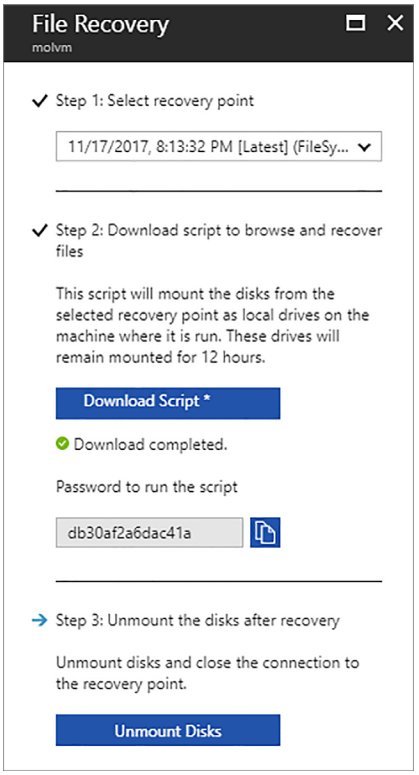


Рис. 13.6. При выполнении восстановления на уровне файлов вы выбираете точку, из которой будет выполнено восстановление. Затем на компьютер скачивается скрипт восстановления. Его можно выполнить, только введя сгенерированный пароль. Скрипт восстановления размещает точку восстановления в виде локального тома на вашем компьютере. Восстановив нужные файлы, удалите диски со своего компьютера, и они будут возвращены для использования в хранилище восстановления.

При выполнении скрипта восстановления точка восстановления подключается в качестве локальной файловой системы на вашем компьютере. Для виртуальных машин с Windows создается сценарий PowerShell и подключается локальный том, например F:. Для виртуальных машин с Linux точка восстановления устанавливается в виде диска данных, например /dev/sdc1, в домашнем томе. В обоих случаях сценарий восстановления содержит точную информацию о том, где найти свои файлы.

Завершив восстановление файлов из хранилища восстановления, вернитесь на портал Azure и выберите вариант «Отключить диски». В ходе этого процесса диски отключаются от вашего локального компьютера и возвращаются в хранилище восстановления. Не переживайте, если забудете отключить диски в самый напряженный момент, когда нужно быстро восстановить файлы для производственной виртуальной машины. Azure автоматически отключает все подключенные точки восстановления через 12 часов.

ПОЛНОЕ ВОССТАНОВЛЕНИЕ ВИРТУАЛЬНОЙ МАШИНЫ

При полном восстановлении виртуальной машины создается виртуальная машина, которая затем подключается к виртуальной сети и подключает все виртуальные жесткие диски. Давайте попробуем выполнить полное восстановление виртуальной машины. Поскольку всегда рекомендуется тестировать служебные обновления перед выполнением их в реальной среде, это упражнение по восстановлению ВМ станет хорошей практикой.

Попробуйте сейчас

Чтобы выполнить полное восстановление виртуальной машины, выполните следующие действия:

1. В своей группе ресурсов выберите виртуальную машину, резервное копирование которой вы выполнили в предыдущем упражнении.
2. Выберите параметр «Резервное копирование» в меню ВМ слева. Обзорные сведения о резервном копировании должны содержать информацию о том, что создана точка восстановления, как показано на рисунке 13.7. Если этого не происходит, подождите несколько минут и вернитесь к этому упражнению. Либо просто прочитайте, как должен проходить этот процесс.

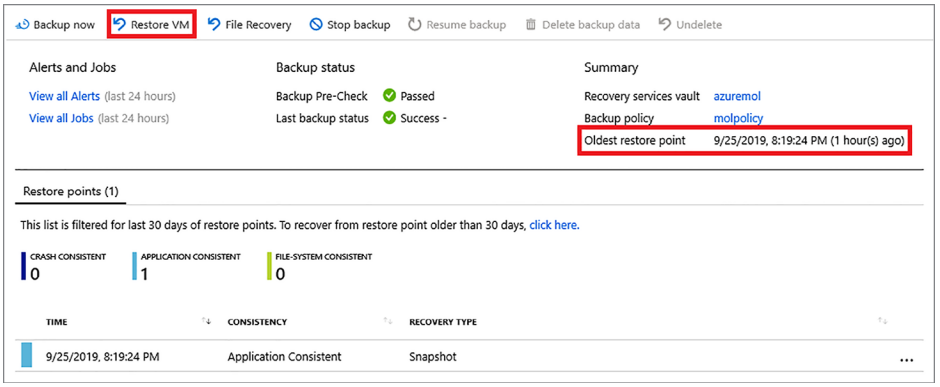


Рис. 13.7. Когда резервное копирование виртуальной машины завершено, на обзорной странице отображаются данные о последнем резервном копировании и доступных точках восстановления. Чтобы начать процесс восстановления, выберите «Восстановить виртуальную машину».

3. Нажмите кнопку «Восстановить виртуальную машину», выберите точку восстановления из списка и нажмите кнопку «ОК».
4. Выберите точку восстановления и способ восстановления ВМ. Вы можете создать новую ВМ или заменить существующую.

   По умолчанию создается новая ВМ. В этой конфигурации создается новая ВМ, она подключается к указанной виртуальной сети, диски восстанавливаются и подключаются.

   Вы также можете заменить существующую ВМ. В этом случае диски восстанавливаются из резервной копии и подключаются к существующей ВМ. Все параметры виртуальной сети или другие параметры конфигурации, примененные к ВМ, сохраняются.

5 Для этого упражнения выберите восстановление в новой ВМ. Укажите имя восстановленной виртуальной машины, например `restoredvm`, а затем проверьте параметры виртуальной сети и хранилища. В производственной среде восстановленная ВМ, как правило, подключается к изолированной виртуальной сети, чтобы это не повлияло на производственный трафик.

6 Нажмите кнопку «OK», а затем — «Восстановить».

Для подключения точки восстановления и создания восстановленной ВМ с ранее подключенными дисками требуется несколько минут. На этом этапе можно подключиться к восстановленной ВМ, чтобы протестировать программные обновления или при необходимости восстановить большие объемы данных.

Кроме того, можно выполнить резервное копирование веб-приложения, так что такой подход работает не только с ВМ. Процедура немного отличается, но принцип тот же. Изменение модели приложения на PaaS (например, веб-приложение) не означает, что можно забыть об основных принципах резервного копирования и хранения данных!

13.2 Azure Site Recovery

Помните, когда мы обсуждали Cosmos DB, вы узнали, что с помощью пары нажатий клавиш можно реплицировать данные в другой регион Azure, тем самым обеспечив избыточность и отказоустойчивость системы? То же самое можно сделать и со всей виртуальной машиной! Azure Site Recovery — это эффективный сервис, возможности которого гораздо шире, нежели репликация виртуальных машин в другой регион. На рисунке 13.8 показано, как Azure Site Recovery координирует рабочие нагрузки между расположениями.

Рис. 13.8. Azure Site Recovery координирует репликацию и перенос физических и виртуальных ресурсов в другое расположение. Локальные расположения и среда Azure могут выполнять функции исходных и целевых точек для защиты, репликации или переноса данных.

Важно отметить, что Azure Site Recovery предназначена не только для виртуальных машин Azure — сервис можно использовать для репликации локальных виртуальных машин VMware или Hyper-V в Azure в целях аварийного восстановления или в рамках переноса данных в Azure. Кроме того, Azure Site Recovery можно использовать исключительно как координатор при репликации локальных виртуальных машин из одного расположения в дополнительное локальное расположение.

Сервис архивации Azure работает не только с Azure. То же самое можно сказать и об Azure Site Recovery — сервис можно использовать не только для репликации виртуальных машин Azure. Сервис архивации Azure и Azure Site Recovery можно использовать как гибридные решения для резервного копирования и аварийного восстановления. Эти сервисы Azure можно использовать для защиты всех рабочих нагрузок: как локальных, так и выполняемых в Azure. Затем можно создать единую структуру отчетности для согласования и проверки — это позволит убедиться в том, что все рабочие нагрузки действительно защищены от потери данных.

Зачем использовать Azure Site Recovery? Как правило, на то есть две причины: репликация и миграция.

Репликация защищает вас в случае отключения всего региона Azure. Весь регион может оказаться не в сети только в случае какого-нибудь катастрофического события, но если вы работаете в сфере ИТ, вы знаете, что возможно все. Даже группы и зоны доступности, о которых мы говорили в главе 7, как правило, защищают только от незначительных отключений в регионе Azure. Если от сети отключится весь регион, отключится и ваше приложение. При использовании Site Recovery вся среда приложений, в том числе ресурсы виртуальной сети, реплицируется в дополнительный регион Azure. Одним нажатием кнопки это дополнительное расположение можно перевести в оперативный режим и активировать. Затем трафик можно направлять в это дополнительное расположение, продолжая обслуживать своих клиентов. На рисунке 13.9 в общих чертах представлена защита среды с помощью Azure Site Recovery.

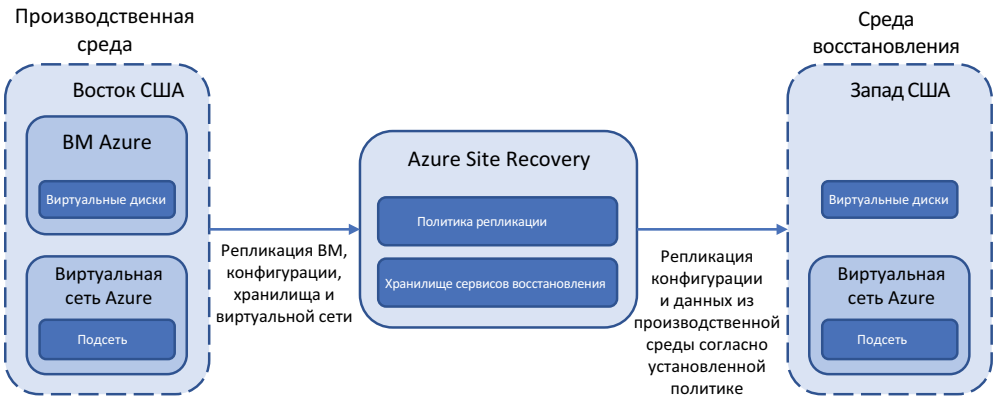


Рис. 13.9. Azure Site Recovery реплицирует конфигурацию, данные и виртуальные сети из производственной среды в среду восстановления. Виртуальные машины не создаются в среде восстановления до тех пор, пока не будет инициирована отработка отказа. Реплицируются только данные.

Виртуальная машина — это всего лишь мета-данные, которые определяют размер виртуальной машины, подключенные диски и сетевые ресурсы, к которым подключается виртуальная машина. Метаданные реплицируются, что позволяет быстро создавать ВМ в случае отработки отказа. Виртуальные диски реплицируются в среду восстановления и подключаются, когда во время отработки отказа создается ВМ восстановления.

Для репликации из среды Azure в другую среду Azure нет определенного расписания. Диски реплицируются практически в режиме реального времени. Когда данные на исходных виртуальных дисках меняются, они реплицируются в среду восстановления. Что касается гибридных рабочих нагрузок, то когда вы обеспечиваете безопасность локальных виртуальных машин VMware или HyperV, вы устанавливаете политики, контролирующие расписание репликации.

Если мы сосредоточимся на репликации между регионами Azure, то как реплицируются данные практически в реальном времени? В расположении производственной среды создается кэш учетной записи хранилища, как показано на рисунке 13.10. Изменения, которые записываются на производственные виртуальные диски, немедленно реплицируются в кэш учетной записи хранения. Затем кэш учетной записи хранения реплицируется в среду восстановления. Этот кэш учетной записи хранения выполняет функцию буфера, так что никакие задержки в репликации, которые происходят в удаленном расположении восстановления, не повлияют на производительность производственной нагрузки.

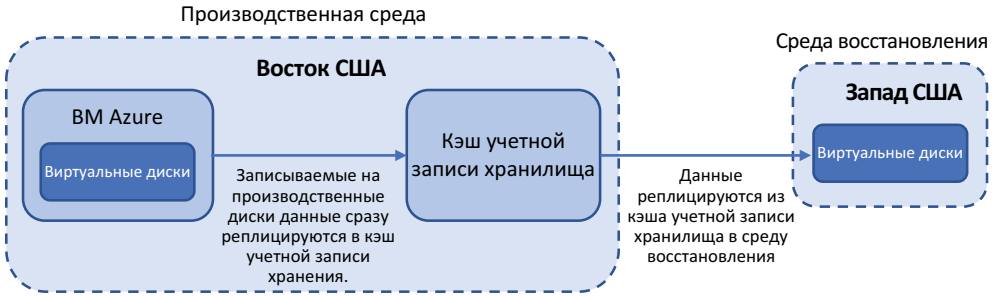


Рис. 13.10. Изменения на производственных дисках немедленно реплицируются в кэш учетной записи хранения. Кэш учетной записи хранения не позволяет изменениям производительности влиять на производственные рабочие нагрузки, пока они ожидают репликации изменений в удаленное расположение восстановления. Изменения в кэше учетной записи хранения затем реплицируются в удаленную точку восстановления с целью сохранения согласованности данных.

Процедура настройки Site Recovery для репликации между регионами Azure очень проста, однако создание всех необходимых реплицированных ресурсов и первоначальная репликация данных занимают определенное время. В практическом упражнении в конце главы вам потребуется настроить репликацию между регионами Azure.

Что можно сделать с виртуальными машинами, реплицированными в дополнительное расположение с помощью Azure Site Recovery? Остается скрестить пальцы и надеяться, что они вам не понадобятся! Однако существует несколько сценариев, когда они необходимы.

Первый, наверное, очевиден: крупное отключение. Если регион Azure становится полностью недоступным, например из-за стихийного бедствия в зоне, можно запустить отработку отказа ресурсов. В этом случае Azure Site Recovery получает команду создать

виртуальные машины в расположении восстановления, используя метаданные реплицированных ВМ, а затем подключить к ним подходящие виртуальные жесткие диски и наладить прочие сетевые соединения. В этой ситуации можно действовать проактивно: например, если прогнозируется стихийное бедствие в регионе Azure, можно запустить отработку отказа *до* того, как это произойдет. При таком подходе необходимо выбрать наиболее подходящее время потенциального простоя, пока ресурсы переводятся в дополнительное расположение. Лучше всего выбрать для этого нерабочее время. После того как минует прогнозируемое бедствие в основном регионе Azure, можно вернуть ресурсы туда и продолжить нормальную работу.

Второй возможный сценарий отработки отказа — тестирование работоспособности процесса. Точно так же как следует регулярно тестировать резервные копии, нужно проверять и разработанный план репликации и отработки отказа. Когда вам на самом деле нужно перевести дополнительное расположение в оперативный режим, будет очень неприятно узнать, что что-то неправильно настроено в виртуальных сетях или отработку отказа какого-либо приложения выполнить невозможно. К счастью, Azure предоставляет возможность тестирования отработки отказа. В качестве дополнительного расположения обычно используется изолированная виртуальная сеть Azure, а производственные рабочие нагрузки по-прежнему выполняются в основном расположении. Если используется Azure Site Recovery, не забудьте регулярно тестировать процедуру отработки отказа.

13.3 Практическое упражнение: настройка виртуальной машины для использования сервиса Site Recovery

К настройке репликации локальных виртуальных машин VMware или Hyper-V с помощью Azure Site Recovery предъявляется ряд требований. Это отличная функция — как для аварийного восстановления, так и для переноса виртуальных машин в Azure. Однако на изучение этой технологии нужно гораздо больше времени, чем обеденный перерыв! Поэтому если вы хотите узнать больше об этих сценариях, перейдите на сайт http://mng.bz/x71V.

Давайте настроим репликацию между регионами Azure, используя тестовую ВМ, которую вы создали и архивировали ранее:

1. На портале Azure в меню слева выберите «Группы ресурсов».
2. Выберите группу ресурсов, использованную в предыдущих упражнениях, например azuremolchapter13.
3. Выберите виртуальную машину, созданную в предыдущих упражнениях, например molvm.
4. Выберите «Аварийное восстановление» в меню слева в окне виртуальной машины.
5. В разделе «Дополнительные параметры» изучите параметры по умолчанию, используемые сервисом Azure Site Recovery, для создания группы ресурсов и виртуальной сети в целевом расположении. Кэш учетной записи хранения создается для репликации с исходных виртуальных дисков, а хранилище и политика сервисов восстановления — для управления процессом репликации.
6. Здесь не нужно ничего менять, хотя если вы используете Site Recovery в производственной среде, где требуется обеспечить защиту большого числа виртуальных машин, необходимо проверить, как виртуальные машины сопоставляются с существующими реплицированным виртуальным сетям и подсетям. Для этого упражнения просмотрите и включите репликацию с использованием значений по умолчанию.

А теперь возвращайтесь к работе. Я серьезно! Нужно достаточно много времени, чтобы настроить все реплицированные ресурсы и завершить первоначальную синхронизацию данных. Не следует дожидаться этого, если только ваш босс не против бесконечных обеденных перерывов!

Защита резервных копий от удаления

Надеюсь, вы удаляли группы ресурсов и соответствующие ресурсы в конце каждой главы, чтобы сохранить бесплатные средства на счете Azure для использования в дальнейшем.

Если безопасность ваших виртуальных машин обеспечивается сервисом архивации Azure или Site Recovery, удалить хранилище сервисов восстановления или группу ресурсов для этой ВМ невозможно. Платформа Azure «знает», что у вас есть архивированные или реплицированные активные данные и блокирует удаление этих ресурсов.

Чтобы удалить защищенные виртуальные машины, сначала необходимо отменить все активные задания резервного копирования и отключить все реплицированные ВМ. После этого можно выбрать: хранить защищенные данные или удалить их. Для практических упражнений в этой главе выберите удаление точек восстановления. В целях безопасности Azure автоматически обратимо удаляет эти точки восстановления и позволяет восстановить их в течение 14 дней. Здесь ничего не нужно настраивать, и вы не можете принудительно удалить эти точки восстановления. Вы также можете отключить функцию обратимого удаления хранилища сервисов восстановления, выбрав «Свойства» для хранилища на портале Azure, но я не рекомендую это делать.

Хорошая новость в том, что остальную часть группы ресурсов можно удалить, и вы ничего не платите за эти обратимо удаленные точки восстановления. После истечения 14-дневного периода обратимого удаления хранилище сервисов восстановления можно удалить как обычно. Цель в том, чтобы защитить вас от случайного или злонамеренного удаления точек восстановления и дать время, чтобы понять, нужны ли они на самом деле, и восстановить их.

14 Шифрование данных

Безопасность ваших данных важна, но еще важнее безопасность данных ваших клиентов. Не проходит и недели, чтобы в новостях не объявили об утечке данных какой-либо крупной компании. Часто эти инциденты вызваны недостатками в системе безопасности, неверной конфигурацией или простой небрежностью. В цифровую эпоху злоумышленники могут легко автоматизировать свои попытки получения доступа к вашим данным. Время, потраченное на ликвидацию инцидента безопасности на уровне приложений, не сравнится с тем, как долго компания будет возвращать себе доверие клиентов, если *их* данные попадут в руки злоумышленников.

Azure предоставляет функции шифрования, так что теперь сложно будет заявить, что у вас нет времени или знаний на обеспечение безопасности ваших данных. В этой главе мы рассматриваем шифрование данных, которые хранятся в хранилище Azure, на управляемых дисках или на виртуальной машине. О шифровании данных написано много книг, и в этой главе я не ставлю перед собой цель углубиться в методы шифрования и другие аспекты этой технологии. Мне бы хотелось показать вам, как включать некоторые базовые функции и сервисы Azure, чтобы обеспечить безопасность ваших данных на протяжении всего жизненного цикла приложений.

14.1 Что такое шифрование данных?

Вы замечали, что когда вы приобретаете что-либо в Интернете в адресной строке изображен небольшой значок замка, указывающий, что на веб-сайте используется протокол HTTPS? Почему настоятельно не рекомендуется отправлять сведения о своих банковских картах по обычному, незащищенному HTTP-подключению? Каждый бит данных в пересылаемом между устройствами сетевом пакете можно отследить и изучить. На рисунке 14.1 показано, как совершение покупок в Интернете не по безопасному HTTPS-подключению может отрицательно сказаться на вашей банковской выписке.

Нет ничего, что оправдывало бы использование небезопасных подключений на веб-серверах. К каждому веб-приложению, которое вы создаете в Azure, автоматически применяется групповой SSL-сертификат.

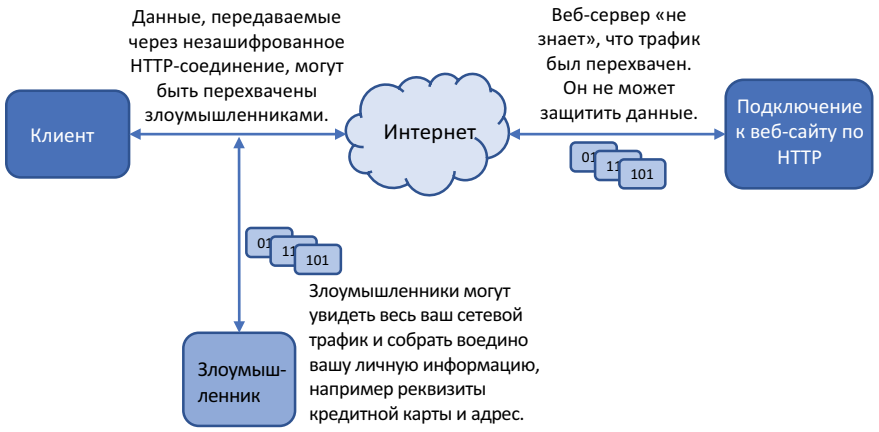


Рис. 14.1. В этом простом примере злоумышленник мог перехватить сетевой трафик, отправляемый по незашифрованному HTTP-подключению. Поскольку ваши данные не зашифрованы, злоумышленник может объединить сетевые пакеты и получить ваши личные и финансовые данные. Если бы вы подключились к веб-серверу по зашифрованному подключению HTTPS, злоумышленник бы не смог прочитать содержимое сетевых пакетов и просмотреть данные.

SSL-сертификат — это цифровой компонент, используемый для обеспечения безопасности веб-сервера и позволяющий веб-браузеру проверять подключение. Групповой SSL-сертификат может использоваться во всем домене, например *.azurewebsites.net — домене по умолчанию для веб-приложений. При создании веб-приложения в главе 3 можно было добавить к веб-адресу префикс https:// to the web address and started to use encrypted communications with your web apps. Это все, что от вас требуется!

Настраиваемые SSL-сертификаты относительно дешевы и просты в использовании. В рамках таких проектов, как Let's Encrypt (https://letsencrypt.org), можно бесплатно получить сертификат и автоматически настроить веб-сервер за считаные минуты. Вы также можете приобрести и использовать сертификат App Service, который напрямую интегрируется с веб-приложениями. Сертификаты App Service хранятся в сервисе Azure Key Vault, о чем мы подробнее поговорим в главе 15.

При проектировании и создании приложений в Azure следует всегда, когда только возможно, обеспечивать безопасный обмен данными. Такой подход помогает защитить передаваемые данные, а как насчет того, когда эти данные записываются на диск? Аналогичный процесс существует для дисков и виртуальных машин — он обеспечивает безопасность ваших данных в покое. На рисунке 14.2 показано, как работает шифрование дисков и виртуальных машин.

Надеюсь, что эти упрощенные примеры шифрования данных в Azure мотивируют вас использовать шифрование при проектировании и создании приложений в Azure. Большинство клиентов рассчитывают на безопасность своих данных, шифровать данные требуют и многие законодательные и нормативные требования, регулирующие деятельность компаний. Следует думать не только о возможных штрафах за нарушение безопасности данных или потере доверия клиентов. Только подумайте, как утечка персональных и финансовых данных клиентов повлияет на их повседневную жизнь! Вероятно, вам не понравится и мысль об утечке собственных данных, поэтому делайте все возможное для защиты данных своих клиентов.

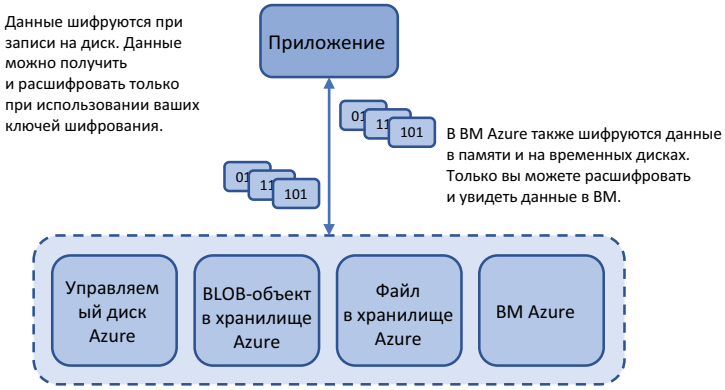


Рис. 14.2. Если вы зашифровали данные, расшифровать их и просмотреть содержимое можете только вы. Если бы злоумышленник получил доступ к виртуальному диску или отдельным файлам, он бы не смог расшифровать содержимое. Методы шифрования можно сочетать: клиенты могут подключаться к вашему веб-ресурсу по HTTPS, вы можете настроить принудительную передачу трафика в учетные записи хранилища по HTTPS, а затем зашифровать данные, которые записываются на диск.

14.2 Шифрование данных во время хранения

Если шифрование данных так важно, как использовать эту возможность в Azure? Просто продолжайте делать все то, о чем я рассказал вам в этой книге! В самом начале я упомянул, что на всех виртуальных машинах необходимо использовать управляемые диски, так? На то есть множество причин, и одна из них — безопасность. Управляемый диск шифруется автоматически. Вам не нужно ничего настраивать, а включение шифрования не влияет на производительность. Отключить эту функцию невозможно — ваши данные автоматически шифруются, когда хранятся на управляемых дисках.

Что означает термин *шифрование хранимых данных*? При использовании управляемых дисков данные шифруются во время записи в базовое хранилище Azure. Данные, которые хранятся на временных дисках, а также данные, которые существуют в памяти виртуальной машины, не шифруются. Только когда данные ОС или диска данных *попадают* на соответствующий физический диск хранятся на нем), они шифруются. На рисунке 14.3 показано шифрование данных при записи на управляемый диск.

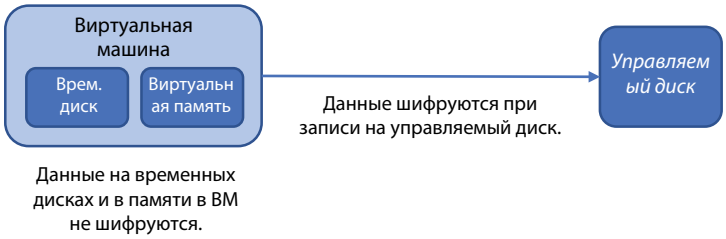


Рис. 14.3. Данные шифруются, когда записываются на управляемый диск. Данные в памяти виртуальной машины или данные на локальных временных дисках виртуальной машины шифруются только в том случае, если включено шифрование всех виртуальных машин (об этом мы поговорим в разделе 14.4.2). Автоматическое шифрование данных, записываемых на управляемые диски, не увеличивает нагрузку на ВМ. Платформа Azure выполняет шифрование в своем хранилище. Виртуальной машине не требуется заниматься шифрованием и расшифровкой.

Шифрование хранимых данных на управляемых дисках означает, что это шифрование никак не влияет на производительность виртуальных машин. Виртуальной машине не требуется выполнять дополнительные вычислительные операции для шифрования или расшифровки данных, так что все ресурсы ЦП можно использовать для работы

приложений. В стандартных сценариях шифрования ВМ виртуальная машина тратит определенный объем вычислительных ресурсов для шифрования данных и управления этим процессом. Единственным недостатком автоматического шифрования управляемых дисков является тот факт, что защищены только диски ОС и диски данных. Под угрозой могут оказаться данные в памяти или на временных дисках ВМ.

Microsoft управляет цифровыми ключами шифрования на платформе Azure, осуществляя автоматическое шифрование управляемых дисков. Это имеет еще одно преимущество: вы можете автоматически зашифровать свои данные без необходимости создавать, передавать, отзывать ключи или управлять ими — вы делегируете обязанность по защите этих ключей Microsoft.

14.3 Шифрование сервиса хранилища

Автоматическое шифрование управляемых дисков — это прекрасно, но что если для файлового хранилища или хранилища BLOB-объектов используется хранилище Azure? Шифрование сервиса хранилища Azure (SSE) позволяет шифровать данные на уровне учетной записи хранения. Данные шифруются во время сохранения в учетной записи. Опять же, Microsoft берет на себя все обязанности, связанные с ключами шифрования, так что вам не нужно ничего настраивать и ничем управлять. Платформа Azure берет на себя создание ключей и управление ими. Вы можете создать и использовать собственные ключи шифрования — это потребует небольших дополнительных усилий по управлению. Шифрование хранилища Azure, как и автоматическое шифрование неактивных управляемых дисков, включается автоматически при создании учетной записи.

Цель автоматического шифрования управляемых дисков и SSE — максимально облегчить для вас шифрование данных и предоставить вам возможность уделить время проектированию, разработке и работе своих приложений. На рисунке 14.4 показано, как SSE защищает ваши данные и обеспечивает принудительный безопасный обмен данными при передаче данных.

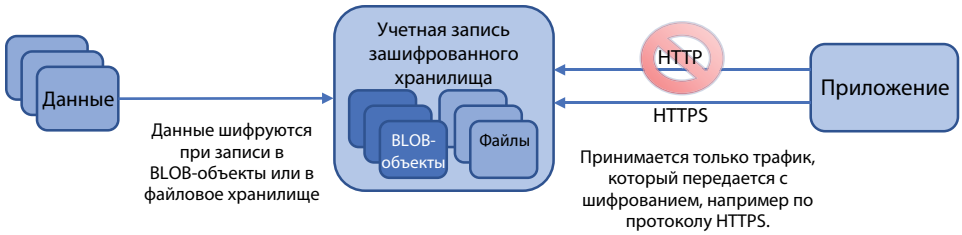


Рис. 14.4. Если включено шифрование SSE, BLOB-объекты и файлы Azure шифруются при записи данных на диск. Таблицы и очереди Azure не шифруются. В целях обеспечения дополнительной безопасности данных можно настроить принудительное использование защищенных протоколов связи, таких как HTTPS, для любых операций обмена данными с учетной записью хранилища. Это защитит передаваемые данные до того момента, как они будут зашифрованы на диске.

Принудительное использование безопасной передачи трафика хранилища

Наряду с включением SSE можно настроить принудительное использование безопасного метода обмена данными для всех запросов и передач в хранилище. Если эта настройка задана, все вызовы API REST будут использовать протокол HTTPS, а все файловые подключения Azure, для которых не включено шифрование (например, старые версии протокола SMB), будут удаляться.

(продолжение)

В наборах SDK Azure, таких как примеры с Python, которые мы рассматривали в четвертой главе, могут использоваться зашифрованные подключения. В справочной документации к набору SDK для каждого языка содержатся инструкции по настройке безопасного обмена данными.

Механизмы безопасного обмена данными необходимо интегрировать в приложения еще на этапе разработки. Если какие-либо компоненты не были изначально настроены должным образом, при попытке включить безопасный обмен данными в существующем приложении могут возникнуть проблемы. По крайней мере необходимо сначала протестировать безопасный обмен данными в существующем приложении в среде разработки.

Попробуйте сейчас

Чтобы создать учетную запись хранилища и включить шифрование и безопасный обмен данными, выполните следующие действия:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Создайте группу ресурсов, введите имя, например `azuremolchapter14`, и расположение, например `eastus`:

```
az group create --name azuremolchapter14 --location eastus
```

3. Создайте учетную запись хранения, выполнив команду `az storage account create`. Укажите уникальное имя, например `azuremolstorage`, и введите группу ресурсов, созданную на шаге 2. Введите тип учетной записи хранения, например `Standard_LRS` для локально избыточного хранилища. Чтобы настроить принудительный безопасный обмен данными, настройте параметр `--https-only`.

```
az storage account create \
  --name azuremolstorage \
  --resource-group azuremolchapter14 \
  --sku standard_lrs \
  --https-only true
```

4. Убедитесь, что учетная запись хранения зашифрована и включена для безопасного обмена данными, запросив `enableHttpsTrafficOnly` и параметры шифрования:

```
az storage account show \
  --name azuremolstorage \
  --resource-group azuremolchapter14 \
  --query [enableHttpsTrafficOnly,encryption]
```

Выходные данные подобны следующим:

```
[
  true,
  {
    "keySource": "Microsoft.Storage",
    "keyVaultProperties": null,
    "services": {
      "blob": {
```

```
        "enabled": true,
        "lastEnabledTime": "2019-09-27T03:33:17.441971+00:00"
      },
      "file": {
        "enabled": true,
        "lastEnabledTime": "2019-09-27T03:33:17.441971+00:00"
      },
      "queue": null,
      "table": null
    }
  }
]
```

14.4 Шифрование виртуальной машины

Автоматическое шифрование управляемых дисков Azure обеспечивает определенный уровень безопасности виртуальной машины. Чтобы обеспечить комплексную безопасность данных на виртуальной машине, можно зашифровать саму виртуальную машину. Этот процесс подразумевает не только шифрование соответствующих виртуальных жестких дисков. Диск ОС и все подключенные диски данных (вместе с временным диском) шифруются. Память виртуальной машины также шифруется, что уменьшает поверхность атаки еще больше. Для шифрования ВМ используются цифровые лючи.

Одним из преимуществ шифрования всей ВМ является возможность управлять ключами шифрования. Эти ключи шифрования безопасно хранятся в хранилище ключей Azure, и вы можете выбирать тип ключей для использования: программный или аппаратный. Вы контролируете эти ключи, поэтому можете определить условия доступа к ним и использовать инструменты управления доступом на основе ролей и средства аудита для отслеживания использования. Кроме того, можно менять ключи шифрования по заданному расписанию — аналогично смене пароля каждые 60–90 дней. Дополнительные инструменты контроля и задачи по управлению ключами шифрования увеличивают вашу нагрузку в области управления, но гарантируют максимальную гибкость в обеспечении безопасности ваших данных. Кроме того, они могут потребоваться для выполнения требований закона. Рассмотрим Azure Key Vault подробнее.

14.4.1 Хранение ключей шифрования в Azure Key Vault

Глава 15 посвящена сервису Azure Key Vault, но прямо сейчас мне бы хотелось продемонстрировать вам возможности шифрования данных и виртуальных машин. Буду краток: Azure Key Vault — это цифровое хранилище для безопасного хранения ключей шифрования, SSL-сертификатов и секретных сведений (например, паролей). В целях обеспечения избыточности хранилища ключей реплицируются в регионы Azure. Подобная репликация защищает ваши ключи и секретные сведения и гарантирует их доступность для использования.

Доступ к вашим хранилищам ключей есть только у вас. Вы создаете и сохраняете объекты в хранилищах ключей, а затем определяете, кто имеет доступ к этим хранилищам. Microsoft управляет соответствующим сервисом Key Vault, но не имеет доступа к содержимому хранилищ. Подобное разграничение безопасности приводит к тому, что когда вы шифруете данные в Azure, расшифровать и просмотреть их можете только вы.

Попробуйте сейчас

Чтобы создать хранилище ключей и ключ шифрования, выполните следующие действия:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Создайте хранилище ключей с помощью команды `az keyvault create`. Укажите группу ресурсов, которую вы создали в предыдущем упражнении, например azuremolchapter14, и укажите уникальное имя для своего хранилища ключей, например `azuremolkeyvault`:

```
az keyvault create \
  --resource-group azuremolchapter14 \
  --name azuremolkeyvault \
  --enabled-for-disk-encryption
```

Давайте подумаем о том, зачем добавлять параметр для команды `--enabled-for-disk-encryption`. Когда вы шифруете виртуальную машину, платформа Azure должна иметь возможность запустить и расшифровать виртуальную машину, чтобы обеспечить ее работу. У платформы Azure нет никаких разрешений на доступ к этим данным, у Microsoft нет доступа к просмотру и использованию ключей шифрования ни для чего, кроме запуска виртуальной машины. Когда вы включаете хранилище ключей для шифрования дисков, вы предоставляете Azure разрешения на доступ к хранилищу ключей и использование ключа шифрования, связанного с виртуальной машиной.

Повторюсь, у Microsoft нет доступа к этим ключам или вашим данным — только полномочия, необходимые для запуска зашифрованной виртуальной машины. Достаточно сложно сделать что-то с зашифрованной виртуальной машиной, если ее невозможно загрузить. На рисунке 14.5 показано, как платформа Azure использует ключ шифрования для запуска зашифрованной виртуальной машины.

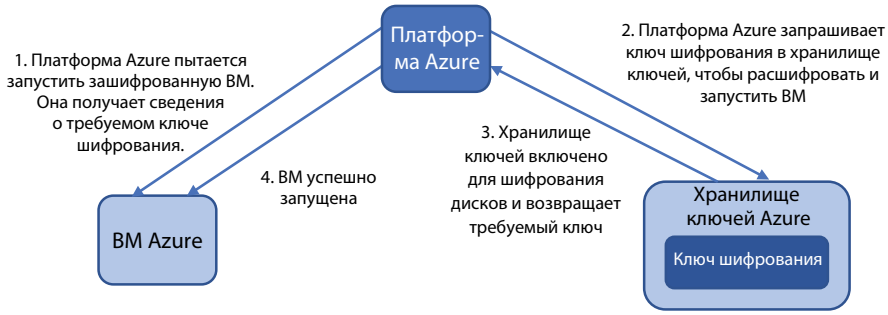


Рис. 14.5. Если хранилище ключей включено для шифрования дисков, оно предоставляет платформе Azure разрешение запрашивать и использовать ключ шифрования для успешного запуска зашифрованной виртуальной машины.

Ключи можно создавать и сохранять в программном обеспечении или аппаратных модулях безопасности (HSM) для обеспечения дополнительной безопасности. Программные ключи отлично подходят для решения многих задач, однако ввиду действующих в организации требований безопасности может понадобиться использовать аппаратные модули безопасности. Об этом подробнее говорится в главе 15.

3. Чтобы создать ключ, укажите хранилище, созданное на шаге 2, например azuremolkeyvault, а затем укажите имя ключа, например `azuremolencryptionkey`:

```
az keyvault key create \
  --vault-name azuremolkeyvault \
  --name azuremolencryptionkey \
  --protection software
```

14.4.2 Шифрование виртуальной машины Azure

Ключ шифрования, созданный в разделе 14.4.1, можно использовать для шифрования многих ВМ. Это снижает нагрузку по управлению ключами и, если используются масштабируемые наборы ВМ, позволяет автоматически масштабировать число экземпляров ВМ без необходимости всякий раз создавать ключи шифрования. Или же у каждой ВМ может быть собственный ключ шифрования, что делает процесс сложнее, но обеспечивает дополнительный уровень безопасности для виртуальных машин. Например, если один и тот же ключ шифрования используется для серверных ВМ и ВМ баз данных, то теоретически злоумышленник с этим ключом может получить доступ к данным обоих наборов ВМ. Если применяются разные ключи, число потенциально скомпрометированных ВМ будет меньше. В практическом упражнении в конце главы вы зашифруете одну ВМ. Этот процесс подойдет и для масштабируемого набора с несколькими ВМ, но при этом используется только один ключ. Не забудьте спроектировать и интегрировать в приложение функции безопасности, особенно если речь идет о больших приложениях с возможностью автоматического масштабирования.

Когда вы шифруете виртуальную машину, устанавливается расширение Azure VM. Это расширение контролирует шифрование диска ОС, временного диска и подключенного диска данных, а также данные в памяти, как показано на рисунке 14.6. Для виртуальных машин с Windows используется механизм шифрования BitLocker. Для виртуальных машин с Linux для шифрования используется механизм dm-crypt. Расширение виртуальной машины может сообщить о состоянии шифрования и при необходимости расшифровать виртуальную машину.

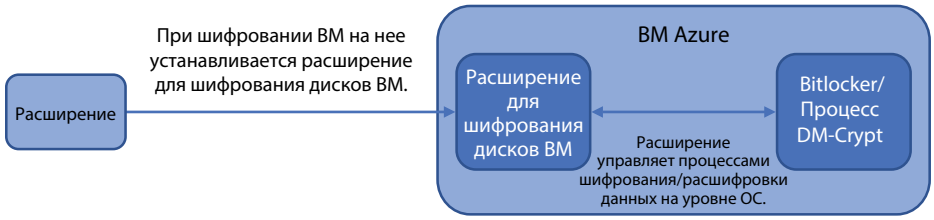


Рис. 14.6. При шифровании ВМ устанавливается расширение для шифрования дисков Azure. Это расширение управляет использованием BitLocker на ВМ с Windows или использованием механизма dm-crypt на ВМ с Linux, чтобы обеспечить возможность шифрования данных на вашей виртуальной машине. Это расширение также используется, когда вы запрашиваете состояние шифрования виртуальной машины.

Поскольку расширение шифрования дисков ВМ использует BitLocker или dm-crypt, существуют определенные ограничения на использование шифрования ВМ. Большинство образов в Azure Marketplace поддерживают шифрование дисков, несмотря на то что налагаются определенные ограничения на размеры ВМ с целью обеспечения шифрования или расшифровки подключенных сетевых папок, например с файлами Azure. Наиболее

полная информация о поддерживаемых ограничениях и других аспектах шифрования ВМ доступна в актуальной документации Azure по адресу http://mng.bz/yyvd.

В этой главе представлен краткий обзор безопасности данных и функций шифрования в Azure. Автоматическое шифрование управляемых дисков и SSE не требует сложной настройки, поэтому ничто, по сути, не мешает вам их использовать.

14.5 Практическое упражнение: шифрование виртуальной машины

Изучить все эти возможности в действии можно, зашифровав виртуальную машину с помощью ключа шифрования, который вы сохранили в хранилище ключей:

1. Создайте виртуальную машину. Большинство образов Linux, доступных в Azure Marketplace, поддерживают шифрование, равно как и образы Windows Server (начиная с Server 2008 R2 и выше). Чтобы упростить и ускорить процесс, создайте виртуальную машину LTS Ubuntu — точно так же, как почти везде в этой книге: Так как ВМ требуется достаточно памяти для шифрования диска, укажите размер Standard_D2s_v3:

```
az vm create \
  --resource-group azuremolchapter14 \
  --name molvm \
  --image ubuntults \
  --size Standard_D2s_v3 \
  --admin-username azuremol \
  --generate-ssh-keys
```

2. Включите шифрование для ВМ, укажите имя хранилища Azure Key Vault и цифровой ключ, созданный в предыдущем упражнении:

```
az vm encryption enable \
  --resource-group azuremolchapter14 \
  --name molvm \
  --disk-encryption-keyvault azuremolkeyvault \
  --key-encryption-key azuremolencryptionkey
```

   Нужно несколько минут, чтобы установить расширение шифрования дисков на виртуальных машинах Azure и начать процедуру шифрования виртуальной машины.

3. Когда начнется шифрование, отслеживайте прогресс и будьте готовы перезапустить виртуальную машину, чтобы завершить шифрование. Проверьте состояние следующим образом:

```
az vm encryption show \
--resource-group azuremolchapter14 \
--name molvm \
--query 'status'
```

   Вот пример выходных данных виртуальной машины в процессе шифрования. В начале сообщение о состоянии отображается так:

```
[
  {
    "code": "ProvisioningState/succeeded",
    "displayStatus": "Provisioning succeeded",
    "level": "Info",
    "message": "OS disk encryption started",
    "time": null
  }
]
```

Шифрование диска может занять некоторое время, поэтому рекомендую вам вернуться к этому упражнению где-то через час, если, конечно, ваш обед не бесконечен! Конечно, я *все еще* не ваш босс, но разве не скучно смотреть на одно и то же сообщение о статусе шифрования?

4 Когда статус шифрования поменяется на Шифрование выполнено успешно для всех томов, перезапустите ВМ:

```
az vm restart --resource-group azuremolchapter14 --name molvm
```

Можно опять проверить состояние шифрования виртуальной машины, выполнив команду `az vm encryption show`, чтобы подтвердить, что ВМ зашифрована.

Помните о своих обязанностях по дому

У вас не ушло много времени на два последних практических упражнения в конце главы, однако их завершения вам пришлось ждать довольно долго. Не забудьте удалить ресурсы, как только перестанете с ними работать.

Помните, что в главе 13 вам нужно отключить защиту сервиса архивации Azure или Site Recovery, чтобы удалить хранилище и группу ресурсов сервисов восстановления (по истечении 14-дневного периода обратимого удаления точек восстановления). Не забудьте удалить ресурсы, использованные вами для выполнения практических упражнений, чтобы не тратить на них слишком много средств на счете Azure.

15 Защита информации с помощью Azure Key Vault

Почти каждую неделю появляются новости об инцидентах, связанных с нарушением кибербезопасности в крупных компаниях. Точно так же как вы используете разные формы автоматизации для роста и репликации своих приложений и данных, злоумышленники автоматизируют собственные действия. Маловероятно, что кто-либо постарается атаковать вашу систему вручную. Это осложняет защиту ваших систем 24 часа в день, 7 дней в неделю, 365 (или 366, хорошо!) дней в году.

В главе 14 рассматривается шифрование ваших данных и виртуальных машин. Это отличный первый шаг, и мы кратко рассмотрели, как создавать и использовать ключи шифрования, хранящиеся в сервисе Azure Key Vault. Безопасные данные, такие как ключи, секреты и сертификаты, лучше всего хранить в цифровом хранилище, например в хранилище ключей, с возможностью централизованного выпуска, аудита использования критически важных учетных и прочих данных и управления ими. Поскольку приложениям и сервисам требуется доступ к разным ресурсам, они могут автоматически запрашивать, извлекать и использовать эти ключи, секреты и учетные данные. В этой главе вы узнаете, зачем и как создавать безопасное хранилище ключей, контролировать доступ, а затем сохранять и извлекать секреты и сертификаты.

15.1 Защита информации в облаке

По мере того как приложения становятся все сложнее и угроза кибератак возрастает, безопасность становится ключевым аспектом при проектировании и выполнении сервисов. Минимизация риска несанкционированного доступа данных должна стать вашим приоритетом при проектировании программных продуктов, особенно интернет-приложений — как локальных, так и облачных. Вы можете создать лучший онлайн-магазин пиццы в мире, но в этом не будет смысла, если клиенты не доверяют вам свои платежные данные или личные сведения.

Как правило, безопасность приложений и сервисов обеспечивается цифровыми ключами, секретами и сертификатами, как показано на рисунке 15.1. Вместо того чтобы использовать имя пользователя и пароль, которые приходится вводить вручную

множество раз (или, что еще хуже, записывать в незашифрованном файле конфигурации), для сохранения защищенных учетных и прочих данных можно использовать цифровое хранилище. Если приложению или сервису требуется доступ, они запрашивают определенный ключ или секрет, который им нужен, и при этом создается аудиторский след, который позволяет отслеживать злоупотребление инструментами безопасности или нарушения безопасности.

Рис. 15.1. Azure Key Vault — это безопасный способ сохранения цифровой информации, такой как сертификаты, ключи и секреты. Затем эти элементы безопасности доступны непосредственно в приложениях или сервисах, а также ресурсах Azure, таких как виртуальные машины. При минимальном участии человека можно централизованно распространять защищенные учетные данные и сертификаты в своей среде приложений.

Если цифровые хранилища правильно спроектированы и реализованы, их работа почти полностью автоматизирована и безопасна. Службы могут запросить новый сертификат, получить сертификат, который надежно хранится в хранилище, и использовать его для авторизации в других компонентах приложений. Серверы могут настраивать программное обеспечение, получая секреты (например, пароли) из цифрового хранилища, а затем устанавливая компоненты приложения, не сохраняя учетные данные в текстовом файле конфигурации. Администратор приложения может централизованно управлять всеми секретами, ключами и сертификатами сервиса, а также регулярно обновлять их по мере необходимости.

Azure Key Vault предоставляет все эти функции цифровой безопасности и позволяет строго контролировать, какие пользователи и ресурсы могут получать доступ к защищенным данным. Хранилища ключей можно безопасно реплицировать для обеспечения избыточности и повышения производительности приложений, а также интегрировать с распространенными ресурсами Azure, такими как виртуальные машины, веб-приложения и учетные записи хранения Azure.

15.1.1 Программные хранилища и аппаратные модули безопасности

Прежде чем мы начнем рассматривать на практике создание и использование хранилища ключей, важно понять, как ваша защищенная информация хранится в хранилище. Как показано на рисунке 15.2, все ключи, секреты и сертификаты в хранилище ключей хранятся в аппаратном модуле безопасности (HSM). Эти устройства не являются уникальным компонентом Azure — это распространенные в отрасли аппаратные устройства, обеспечивающие высокий уровень безопасности любых сохраненных на них данных.

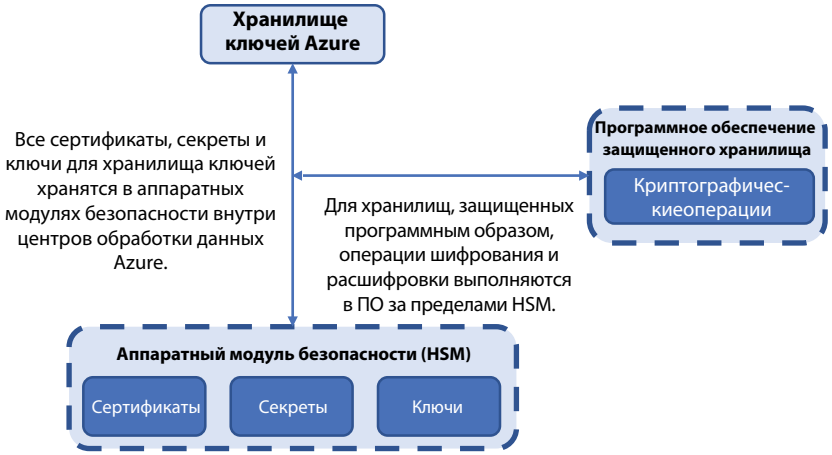


Рис. 15.2. Azure Key Vault — это логический ресурс Azure, однако все сертификаты, секреты и ключи хранятся в аппаратном модуле безопасности. В сценариях разработки и тестирования можно использовать хранилище с программной защитой, которое выполняет все операции шифрования (шифрование и расшифровку данных) в программном обеспечении, а не на аппаратном уровне HSM. В производственной среде следует использовать защищенное HSM хранилище, где все операции обработки выполняются на оборудовании.

Сейчас доступны 2 типа хранилища ключей: с программной и аппаратной защитой. Разобраться в них может быть нелегко, поэтому я хочу прояснить разницу, прежде чем мы начнем с ними работать:

- В *хранилище с аппаратной защитой* хранятся ключи, секреты и сертификаты (внутри аппаратного модуля безопасности), однако все необходимые операции по шифрованию и расшифровке содержимого выполняются платформой Azure в программном обеспечении. Программные хранилища отлично подходят для сценариев разработки и тестирования, хотя вы можете решить, что производственные рабочие нагрузки требуют способа выполнения криптографических операций понадежнее.
- Повторюсь, *в хранилище с аппаратной защитой* хранятся ключи, секреты и сертификаты, однако все необходимые криптографические операции (шифрования и расшифровки содержимого) выполняются непосредственно в HSM. В локальной системе HSM можно создать собственные ключи безопасности, а затем импортировать их в Azure. Существуют дополнительные механизмы и процессы, которые требуется соблюдать, но при использовании этого способа вы сохраняете полный контроль над ключами и гарантируете, что они никогда не покинут HSM.

  Чтобы повысить безопасность и целостность ваших данных, рекомендуется использовать для производственных рабочих нагрузок хранилища с аппаратной защитой.

Независимо от используемого типа хранилища важно помнить, что все ваши данные безопасно хранятся в аппаратном модуле безопасности, который как минимум соответствует стандарту FIPS 140-2 второго уровня, и у Microsoft нет доступа к вашим ключам или возможности их извлечь. За хранилища с защитой HSM требуется платить дополнительно, поэтому как и со всем в Azure и облачных вычислениях, требуется найти баланс между затратами и риском утечки ваших данных.

15.1.2 Создание хранилища ключей и секретов

Цифровое хранилище — отличная идея, но, возможно, вы не уверены, как пользоваться всеми возможностями Azure Key Vault. Давайте в качестве примера построим базовый сервер, на котором выполняется база данных, например MySQL Server, как показано на рисунке 15.3.

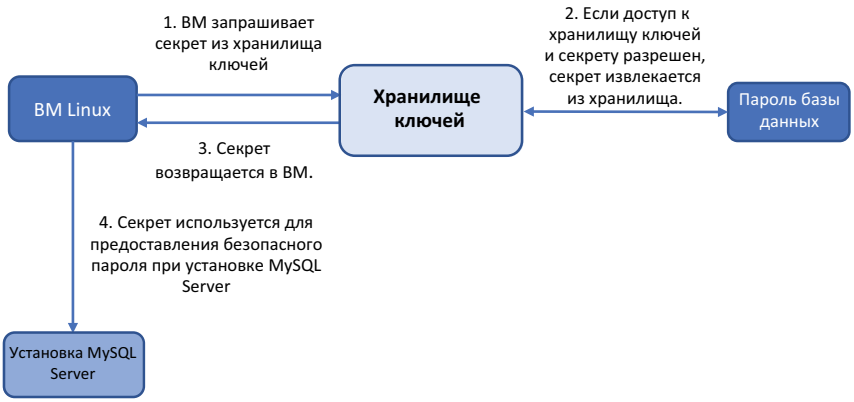


Рис. 15.3. В следующих нескольких упражнениях вы создадите пример секрета, который хранится в хранилище ключей и который можно использовать в качестве пароля базы данных для установки MySQL Server. Создается виртуальная машина с разрешениями запрашивать секрет из хранилища ключей. Извлеченный секрет затем используется для автоматического ввода защищенных учетных данных в процессе установки приложения.

Одним из первых упражнений в начале этой книги было создание виртуальной машины с последующей установкой стека веб-серверов LAMP. Вероятно, вам нужно было ввести пароль MySQL Server, либо автоматически использовался пустой пароль. Теперь, когда вы знаете все о хранилищах ключей, вы можете автоматически получать пароль из хранилища и динамически использовать его для установки и настройки сервера.

Попробуйте сейчас

Чтобы создать хранилище ключей и добавить секрет, выполните следующие действия:

1. Откройте портал Azure, запустите Cloud Shell и создайте группу ресурсов, например `azuremolchapter15`:

```
az group create --name azuremolchapter15 --location eastus
```

2. Создайте хранилище ключей с уникальным именем, таким как `azuremol`, и включите его для развертывания, чтобы можно было использовать хранилище для размещения ключей и сертификатов на виртуальной машине:

```
az keyvault create \
  --resource-group azuremolchapter15 \
  --name azuremol \
  --enable-soft-delete \
  --enabled-for-deployment
```

По умолчанию учетной записи пользователя Azure предоставляется полный доступ к хранилищу ключей со всеми разрешениями. Для целей этих упражнений это хорошо, однако в реальности в целях безопасности следует ограничить доступ к своему хранилищу ключей. Можно добавить параметр `--no-self-perms`, чтобы пропустить назначение разрешений вашей учетной записи.

3 Создайте секрет, например `databasepassword`, и назначьте значение пароля, такое как `SecureP@ssw0rd`. (Очень надежно, не так ли?) Этот секрет можно использовать в качестве учетных данных для сервера баз данных, который развертывается в следующих упражнениях:

```
az keyvault secret set \
  --name databasepassword \
  --vault-name azuremol \
  --description "Database password" \
  --value "SecureP@ssw0rd"
```

4 У вас есть полные права доступа к хранилищу ключей, так что вы можете просмотреть содержимое вашего секрета:

```
az keyvault secret show \
--name databasepassword \
--vault-name azuremol
```

С точки зрения управления можно также выполнять стандартные действия, такие как резервное копирование и восстановление, скачивание, обновление и удаление элементов, сохраненных в хранилище ключей. Одним из дополнительных свойств, настраиваемых при создании хранилища ключей, является параметр `enable-soft-delete`. Если ваши приложения и сервисы не могут извлекать нужные секреты из хранилища ключей, может возникнуть достаточно серьезная проблема в работе приложения. В хранилище ключей могут храниться метаданные секретов сроком до 90 дней с момента фактического удаления, что позволяет восстанавливать данные, которые были удалены по ошибке или злому умыслу.

5 Удалите только что созданный ключ, чтобы моделировать ошибку или, возможно, чей-то злой умысел:

```
az keyvault secret delete \
  --name databasepassword \
  --vault-name azuremol
```

6 Восстановите секрет, чтобы можно было продолжать использовать пароль базы данных со своими приложениями и сервисами:

```
az keyvault secret recover \
  --name databasepassword \
  --vault-name azuremol
```

Если действительно нужно удалить секрет, существует и другой способ безвозвратного удаления секрета. Эта функция позволяет безвозвратно удалить секрет, не дожидаясь истечения 90-дневного периода восстановления по умолчанию.

Используйте команду `az keyvault secret show` еще раз, чтобы просмотреть информацию о своем секрете, и убедитесь, что сохраненный вами пароль действительно находится там после его удаления и восстановления. А сейчас посмотрим, как виртуальная машина может осуществлять доступ к хранилищу ключей и использовать секрет для установки MySQL Server.

15.2 Управляемые удостоверения для ресурсов Azure

Возможность использовать Azure Key Vault для хранения секретов или ключей — это прекрасно, но как осуществлять доступ к этим секретам? Интерфейс командной строки Azure или Azure PowerShell может помочь вам получить доступ к сведениям, хранящимся в хранилище ключей, но зачастую удобнее разрешить виртуальным машинам или приложениям получать нужные секреты или ключи напрямую. Один из способов сделать это — использовать управляемые удостоверения для ресурсов Azure, как показано на рисунке 15.4.

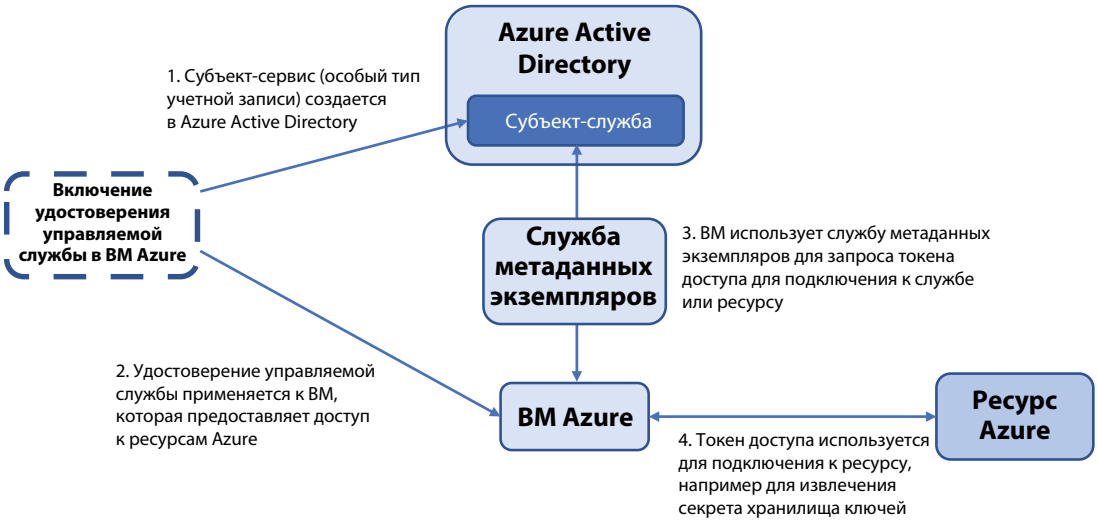


Рис. 15.4. При создании управляемого удостоверения для виртуальной машины в Azure Active Directory создается субъект-сервис. Это особый тип учетной записи, который может использоваться ресурсами для проверки собственной подлинности. Затем виртуальная машина использует конечную точку сервиса метаданных экземпляров для отправки запросов на доступ к ресурсам. Конечная точка подключается к Azure AD для запроса токенов доступа, если виртуальной машине требуется запросить данные у других сервисов. Когда токен доступа возвращается, его можно использовать для запроса доступа к ресурсам Azure, например хранилищу ключей.

Управляемое удостоверение позволяет создать особый тип учетной записи, которую может использовать ресурс Azure, например ВМ. Если вы использовали сервис каталогов, такой как Active Directory, учетная запись компьютера часто используется для идентификации и предоставления доступа к различным сетевым ресурсам, которые требуются компьютеру. Вы не создаете и не используете стандартные учетные записи пользователя для этого типа аутентификации, что повышает уровень безопасности: вы предоставляете ограниченный набор разрешений только компьютеру и можете не беспокоиться о разрешениях пользователей или общем доступе к папкам, например.

Управляемое удостоверение можно сравнить с учетной записью компьютера, но хранится оно в Azure Active Directory (Azure AD). Удостоверение, называемое *субъектом-сервисом*, является уникальным для каждой виртуальной машины и может использоваться для назначения разрешений другим ресурсам Azure, таким как учетная запись хранилища Azure или хранилище ключей. У виртуальной машины есть разрешения на доступ к этим ресурсам, так что вы можете составлять сценарии задач (например, как с сервисом

автоматизации Azure, о чем мы говорим в 18 главе), которые не требуют вмешательства пользователя или ввода имени пользователя и пароля. Виртуальные машины проверяют подлинность самих себя, а платформа Azure разрешает доступ к назначенным ресурсам.

Вы можете создать 2 типа управляемых удостоверений:

- *Назначенный системой* — этот тип управляемого удостоверения применяется напрямую к ресурсу, такому как ВМ, и используется только им. У каждого ресурса есть собственное уникальное удостоверение для аудита или устранения неполадок с доступом. При удалении ресурса управляемое удостоверение удаляется автоматически.
- *Назначенное пользователем* — для указанного управляемого удостоверения создается отдельный ресурс Azure, которым управляет система. Это удостоверение может совместно использоваться другими ресурсами для управления доступом. При удалении любых ресурсов, использующих это удостоверение, оно остается доступным для использования.

Давайте посмотрим, как управляемое удостоверение, назначенное системой, для запроса секрета `databasepassword` в хранилище ключей. Как только виртуальная машина сможет извлечь секрет, пароль можно использовать для автоматической установки сервера базы данных MySQL. Используя хранилище ключей и MSI, можно выполнить пару команд для извлечения секрета из хранилища ключей, выполнения установщика MySQL Server и автоматического предоставления защищенного пароля.

Сервис метаданных экземпляров Azure

Виртуальная машина, включенная с помощью управляемого удостоверения, использует конечную точку REST в сервисе метаданных экземпляров (IMDS), чтобы запросить токен доступа у Azure AD, который затем она может использовать для запроса данных в Azure Key Vault. Но что такое сервис метаданных экземпляров?

IMDS — это конечная точка REST, доступная виртуальным машинам только внутри системы. Конечная точка доступна по адресу, не поддерживающему маршрутизацию, — 169.254.169.254. Виртуальная машина может отправить конечной точке IMDS запрос на извлечение информации о себе, например о регионе Azure или имени группы ресурсов. Эта возможность позволяет виртуальной машине понять, как и где на платформе Azure она используется. Доступ к конечной точке IMDS можно получать на многих разных языках, таких как Python, C#, Go, Java и PowerShell.

В рамках мероприятий по техническому обслуживанию можно также запросить конечную точку IMDS, чтобы виртуальной машине стало известно об ожидающем выполнения событии обновления или перезагрузки. После этого можно выполнить любые требуемые задачи до обновления или перезагрузки. Поскольку IMDS — это конечная точка REST с IP-адресом, не поддерживающим маршрутизацию, не требуется устанавливать агент или расширение для виртуальной машины или беспокоиться о безопасности сети или маршрутизации.

Для управляемых удостоверений конечная точка IMDS используется для ретрансляции запросов на токен доступа к Azure AD. Такой подход позволяет виртуальным машинам запрашивать доступ без необходимости обращаться к Azure AD напрямую.

Попробуйте сейчас

Чтобы создать виртуальную машину с MSI, выполните следующие действия:

1 Создайте ВМ Ubuntu, укажите группу ресурсов, например `azuremolchapter15`, и имя ВМ, например `molvm`. Создается учетная запись пользователя с именем azuremol, а ключи SSH, использованные в предыдущих главах, добавляются в виртуальную машину:

```
az vm create \
  --resource-group azuremolchapter15 \
  --name molvm \
  --image ubuntults \
  --admin-username azuremol \
  --generate-ssh-keys
```

2 В качестве меры обеспечения безопасности не следует разрешать учетным записям доступ ко всем ресурсам в масштабах вашей подписки Azure. Особенно в случае с управляемыми удостоверениями требуется предоставить только минимальные разрешения.

В этом упражнении требуется ограничить доступ только вашей группой ресурсов, например `azuremolchapter15`. Область доступа задается путем запроса идентификатора группы ресурсов командой `--query id`. Затем этот идентификатор присваивается переменной с именем `scope`:

```
scope=$(az group show --resource-group azuremolchapter15
➥--query id --output tsv)
```

3 Создайте управляемое удостоверение, назначенное системой, для виртуальной машины с ролью читателя, чтобы он мог только считывать ресурсы, а не вносить в них изменения. Назначьте областью идентификации группу ресурсов. Предоставляется созданная в предыдущем шаге переменная, которая содержит идентификатор группы ресурсов:

```
az vm identity assign \
  --resource-group azuremolchapter15 \
  --name molvm \
  --role reader \
  --scope $scope
```

4 Примените разрешения в Azure Key Vault, которые предоставляют управляемому удостоверению доступ к субъекту-сервису. Это можно сделать на портале в разделе «Политики доступа» для ресурса Key Vault или с помощью Azure CLI. Мы воспользуемся CLI, чтобы узнать, как получить информацию программно.

Сначала получите информацию о субъекте-сервисе Azure AD для своего управляемого удостоверения. Отфильтруйте данные по `0000000000000 00000` ВМ, созданной на шаге 3, например `molvm`:

```
az ad sp list \
  --display-name molvm \
  --query [].servicePrincipalNames
```

Краткий пример выходных данных приведен ниже. Не беспокойтесь о том, что они означают. Здесь вам потребуется лишь назначить начальные разрешения. Опять же, вы можете использовать портал Azure вместо CLI, если необходимо.

Запомните первое имя `servicePrincipalName`. Это значение используется для назначения разрешений на доступ к ресурсам Azure, таким как хранилище ключей. Оно необходимо на следующем шаге:

```
[
  "887e9665-3c7d-4142-b9a3-c3b3346cd2e2",
  "https://identity.azure.net//
  ➥ihxXtwZEiAeNXU8eED2Ki6FXRPkklthh84S60CiqA4="
]
```

5 Настройте в хранилище ключей политику так, чтобы субъект-сервис для вашей ВМ мог считывать секреты. Затем введите `servicePrincipalName` из шага 4:

```
az keyvault set-policy \
  --name azuremol \
  --secret-permissions get \
  --spn 887e9665-3c7d-4142-b9a3-c3b3346cd2e2
```

Следует отметить, что когда создавалось управляемое удостоверение и его область действия определялась равной группе ресурсов, это не значило, что виртуальная машина могла делать все что угодно. Во-первых, единственная созданная для удостоверения роль — разрешение на чтение для ресурсов. Однако вам по-прежнему требовалось назначить разрешения самому хранилищу ключей. Эти уровни безопасности и разрешения позволяют точно контролировать, к каким ресурсам имеет доступ каждое удостоверение.

Теперь, когда у вас есть доступ к хранилищу ключей, вы наверняка хотите знать, как извлечь секрет, не так ли?

15.3 Получение секрета из виртуальной машины с управляемым удостоверением

Вы сохранили секрет в хранилище ключей для пароля базы данных, а теперь у вас есть виртуальная машина с управляемым удостоверением, которая предоставляет доступ для чтения этого секрета из хранилища ключей. Что теперь? Как извлечь секрет и использовать его? На рисунке 15.5 показано, как виртуальная машина использует IMDS для запроса доступа к ресурсу, например хранилищу ключей. Давайте подробно рассмотрим эти шаги, чтобы понять, как виртуальная машина извлекает секрет.

В большинстве сценариев использования в сервисе Azure Key Vault виртуальная машина не подключается и не извлекает секреты таким образом. Сервис Key Vault особенно эффективен, когда приложения сами извлекают секреты в коде. Код приложения будет использовать соответствующий пакет SDK Azure, например Python, .Net или Java. Чтобы избежать сложностей, связанных с абстрагированием кода, в следующем упражнении используются ВМ и интерфейс командной строки. Работая над этим упражнением, помните, что это обычно происходит в коде приложения.

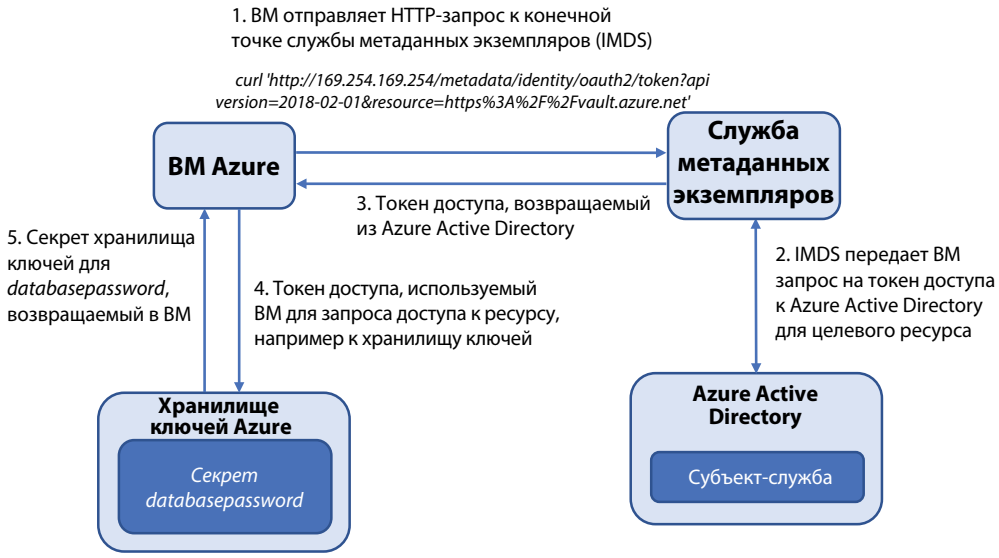


Рис. 15.5. Виртуальная машина использует IMDS для запроса доступа к хранилищу ключей. Конечная точка связывается с Azure AD для запроса токена доступа. Токен доступа возвращается на виртуальную машину, которая затем используется для запроса доступа из хранилища ключей. Если хранилище ключей предоставляет доступ, секрет `databasepassword` возвращается виртуальной машине.

Попробуйте сейчас

Чтобы извлечь и использовать секрет на виртуальной машине с управляемым удостоверением, выполните следующие действия:

1. Получите публичный IP-адрес виртуальной машины, созданный в предыдущем упражнении, например `molvm`:

```
az vm show \
  --resource-group azuremolchapter15 \
  --name molvm \
  --show-details \
  --query [publicIps] \
  --output tsv
```

2. Выполните SSH-подключение к виртуальной машине, например с помощью команды `ssh azuremol@publiclps`.
3. Чтобы получить доступ к хранилищу ключей, требуется токен доступа. Этот токен доступа запрашивается из IMDS. Это простой HTTP-запрос, а на виртуальной машине Linux для отправки этого запроса можно использовать программу `curl`. IMDS передает ваш запрос в AAD:

```
curl 'http://169.254.169.254/metadata/identity/oauth2/token?
➥api-version=2018-02-01&resource=https%3A%2F%2Fvault.azure.net'
➥-H Metadata:true
```

4 Выходные данные немного сложны для понимания, поскольку выглядят как бессмысленный набор символов. Он находится в формате JSON. Чтобы обработать выходные данные в формате JSON и привести их в удобочитаемый вид, установите инструмент синтаксического разбора JSON, который называется `jq`:

```
sudo apt-get update && sudo apt-get -y install jq
```

5 Отправьте запрос `curl` еще раз, но на этот раз просмотрите выходные данные с помощью `jq`:

```
curl 'http://169.254.169.254/metadata/identity/oauth2/token?
➥api-version=2018-02-01&resource=https%3A%2F%2Fvault.azure.net'
➥-H Metadata:true --silent | jq
```

Эти первые несколько шагов помогут понять, как оформляются запросы и как выглядят выходные данные, как показано на рисунке 15.6. Если вы все еще выполняете вход на виртуальную машину и запрашиваете токен доступа вручную, в чем смысл использования управляемого удостоверения? Вы могли бы просто указать свои учетные данные. В продуктивной среде вы бы наверняка использовали сценарий, выполняемый на виртуальной машине для автоматического запроса токена доступа и последующего извлечения секрета из хранилища ключей. Давайте посмотрим, как автоматизировать этот процесс и извлечь секрет.

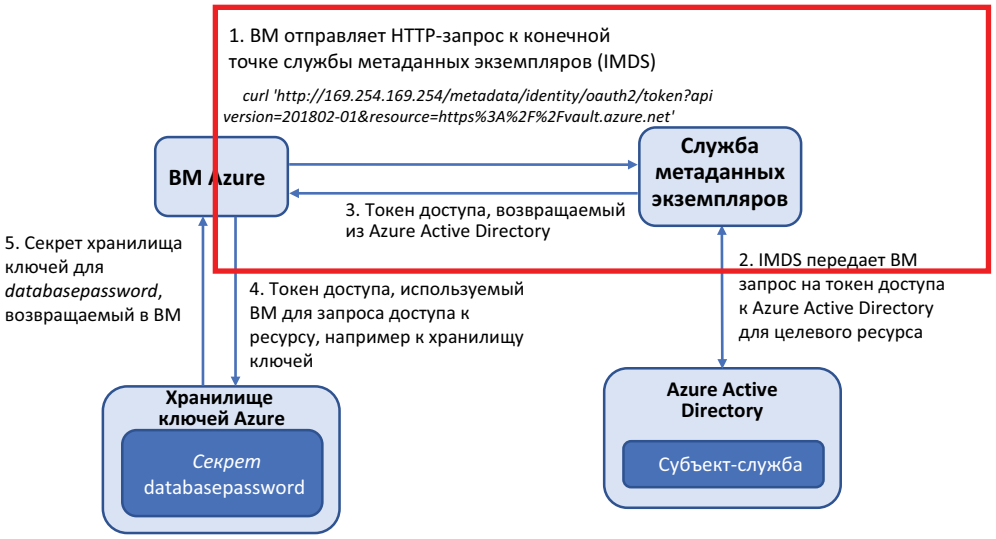


Рис. 15.6. Запрос `curl` охватывает три первые шага на этой схеме. Выполняется запрос `curl`, конечная точка связывается с Azure AD, создается токен доступа.

6 Чтобы упростить (и если бы вы делали все это с помощью сценария), можно использовать программу `jq` для обработки запроса curl, извлечь только токен доступа и задать его в качестве переменной с именем `access_token`:

```
access_token=$(curl
➥'http://169.254.169.254/metadata/identity/oauth2/token?
➥api-version=2018-02-01&resource=https%3A%2F%2Fvault.azure.net'
➥-H Metadata:true --silent | jq -r '.access_token')
```

7 Чтобы помочь понять, как это выглядит, изучите переменную `access_token`:

```
echo $access_token
```

8 А теперь самое интересное! Используйте токен доступа для запроса секрета из хранилища ключей. Давайте выполним всю процедуру вручную, чтобы понять, что происходит.

9 Извлеките секрет, отправив другой запрос `curl`, и форматируйте выходные данные с помощью программы `jq`. Введите собственное имя хранилища ключей в начале адреса https:// address:

```
curl https://azuremol.vault.azure.net/secrets/databasepassword?
➥api-version=2016-10-01 -H "Authorization: Bearer $access_token"
➥--silent | jq
```

Выходные данные имеют примерно следующий вид (показано значение пароля, сохраненного в секрете, а также некоторые дополнительные метаданные о секрете, но пока вам не нужно об этом задумываться):

```
{
  "value": "SecureP@ssw0rd!",
  "contentType": "Database password",
  "id":
➥"https://azuremol.vault.azure.net/secrets/databasepassword/
➥87e79e35f57b41fdb882c367b5c1ffb3",
}
```

Этот запрос `curl` является второй частью рабочего процесса, как показано на рисунке 15.7.

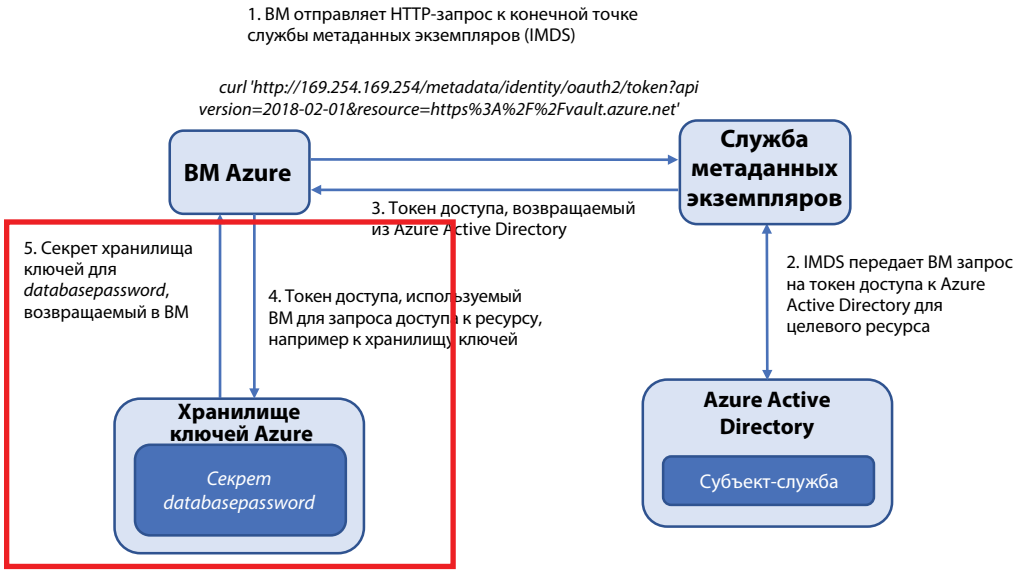


Рис. 15.7. Второй запрос `curl` охватывает последние два шага в схеме. Этот токен доступа используется для запроса секрета из хранилища ключей. Возвращается ответ JSON, который включает значение секрета.

10 Точно так же как вы использовали переменную для сохранения токена доступа, в сценарии тоже можно назначить значение секрета переменной. На этот раз используйте `jq` для обработки ответа, извлеките только секрет значения и задайте его в качестве переменной с именем `database_password`:

```
database_password=$(curl
➥https://azuremol.vault.azure.net/secrets/databasepassword?
➥api-version=2016-10-01 -H "Authorization: Bearer $access_token"
➥--silent | jq -r '.value')
```

11 Опять же, выполним это действие вручную, чтобы вы лучше во всем разобрались: просмотрите содержимое переменной `database_password`:

```
echo $database_password
```

Надеюсь, вы все еще следите за ходом моей мысли! Если вы создаете приложение на языке Python, ASP.NET или Node.js, например, процедура будет напоминать отправку запроса на токен доступа с последующим использованием токена для запроса секрета из хранилища ключей. Вероятно, есть и другие библиотеки, которые можно использовать в своем коде (помимо служебной программы `jq` из командной строки).

В качестве краткого повторения пройденного напомню, что все эти шаги можно сжать до 2 строк кода, как показано в следующем фрагменте.

Фрагмент кода 15.1. Запрос токена доступа и секрета из хранилища ключей

```
access_token=$(curl
➥'http://169.254.169.254/metadata/identity/oauth2/token?
➥api-version=2018-02-01&resource=https%3A%2F%2Fvault.azure.net'
➥-H Metadata:true --silent | jq -r '.access_token')
database_password=$(curl
➥https://azuremol.vault.azure.net/secrets/databasepassword?
➥api-version=2016-10-01 -H "Authorization: Bearer $access_token"
➥-silent | jq -r '.value')
```

Что теперь? Управляемое удостоверение ВМ может извлекать секрет из хранилища ключей. Посмотрим, как можно использовать его удостоверение для установки и настройки MySQL Server.

В Ubuntu можно выбрать параметры конфигурации для установщиков пакетов, таких как MySQL Server. Выбранные параметры конфигурации позволяют предоставлять такие значения, как имена пользователей и пароли, и автоматически использовать их в соответствующей части процесса установки. Запросы на ввод пароля вручную (которые вы, возможно, помните еще со 2 главы) больше не используются.

12 Выберите параметры конфигурации для паролей MySQL Server, используя переменную `database_password`, созданную на шаге 10:

```
sudo debconf-set-selections <<< "mysql-server mysql-server/root_password
➥password $database_password"
sudo debconf-set-selections <<< "mysql-server mysql-
➥server/root_password_again password $database_password"
```

13 Установите MySQL Server. Не отображается никаких запросов, потому что пароль предоставляется выбранными параметрами конфигурации:

```
sudo apt-get -y install mysql-server
```

14 Давайте докажем, что все сработало! Просмотрите переменную `database_password`, чтобы выяснить, каким должен быть ваш пароль:

```
echo $database_password
```

15 Выполните вход в MySQL Server. В ответ на запрос пароля введите значение `database_password` — значение секрета из хранилища ключей:

```
mysql -u root -p
```

Вы выполнили вход в MySQL Server, что подтверждает, что секрет из хранилища ключей был использован для создания учетных данных SQL Server!

16 Введите `exit` дважды, чтобы закрыть командную строку MySQL Server, а затем закройте свой сеанс SSH на виртуальной машине.

Это базовый пример, и вам все равно пришлось бы обеспечивать безопасность MySql Server и предоставлять дополнительные учетные данные для доступа приложений к базам данных или таблицам, например. Преимущество использования секрета из хранилища ключей заключается в гарантии того, что все пароли одинаковы. Если вы используете масштабируемые наборы виртуальных машин, например, каждый экземпляр виртуальной машины может автоматически запросить секрет и установить MySQL Server, чтобы система была готова обслуживать ваши данные приложения. Эти пароли никогда не определяются в сценариях и никто не должен видеть, что это за пароли. Можно даже создавать пароли произвольно и менять их в качестве секретов в хранилище ключей.

Хранение паролей в хранилище ключей — это прекрасно, но можно ли использовать его для хранения сертификатов и автоматического извлечения их из приложений и виртуальных машин? Конечно, да!

15.4 Создание и внедрение сертификатов

Цифровые сертификаты — распространенная форма обеспечения безопасности и аутентификации в веб-сервисах и приложениях. Сертификаты выпускаются центром сертификации, которому (надеюсь!) доверяют конечные пользователи. Сертификат позволяет пользователям убедиться, что веб-сайт или приложение действительно является тем, что заявляет. Всякий раз, когда вы видите веб-сайт с адресом в веб-браузере, который начинается с https:// and has a padlock symbol, the traffic is encrypted an и защищен цифровым сертификатом.

Управление цифровыми сертификатами часто превращается в одну из сложнейших задач управления. Распространенной проблемой является хранение сертификатов и предоставление доступа к ним по мере необходимости в этом сервисов и приложений. В предыдущих упражнениях мы изучили, как можно использовать хранилище ключей для предоставления сервисам и приложениям общего доступа к защищенным секретам и ключам, однако то же самое хранилище ключей может сделать и для сертификатов. Как показано на рисунке 15.8, хранилище ключей можно использовать для запроса, выпуска и хранения сертификатов.

В производственной среде следует всегда использовать для выпуска своих сертификатов доверенный центр сертификации. Для внутреннего использования можно самостоятельно издавать самозаверяющие сертификаты, созданные вами. Другие

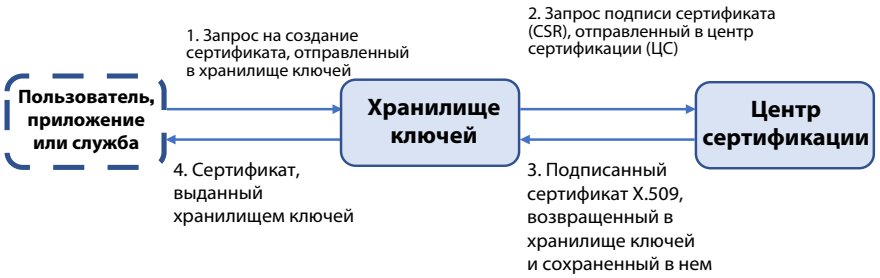


Рис. 15.8. Пользователь, приложение или сервис могут запросить в хранилище ключей новый сертификат. Запрос подписи сертификата (CSR) отправляется хранилищем ключей в сторонний центр сертификации или доверенный внутренний центр сертификации. Azure Key Vault также может функционировать как собственный центр сертификации для создания самозаверяющих сертификатов. Затем центр сертификации выдает подписанный сертификат X.509, который хранится в хранилище ключей. Наконец, хранилище ключей возвращает сертификат исходному запрашивающему.

сервисы и приложения не доверяют таким самозаверяющим сертификатам, так что, как правило, возникает предупреждение, однако самозаверяющие сертификаты позволяют быстро приступить к работе и убедиться, что в работе с зашифрованным трафиком код работает так, как нужно.

Хранилище ключей Azure может создать для вас самозаверяющий сертификат. По сути, Key Vault функционирует как свой собственный центр сертификации, который запрашивает, издает, а затем хранит сертификаты. Давайте воспользуемся этой возможностью для создания самозаверяющего сертификата и посмотрим, насколько легко внедрить его в виртуальную машину. Затем этот сертификат используется для базового веб-сервера, чтобы показать, как быстро включить протокол SSL для защиты вашего интернет-трафика.

Попробуйте сейчас

Чтобы создать и внедрить сертификат в виртуальную машину, выполните следующие действия:

1. Создайте самозаверяющий сертификат в хранилище ключей Azure и введите имя, например `molcert`. Для определения свойств, таких как периоды истечения срока действия, строгость шифрования и формат сертификата, используются политики. Можно создавать различные политики в соответствии с потребностями ваших приложений и сервисов. Для целей этого упражнения используйте политику по умолчанию, которая создает 2048-битный сертификат и действует один год:

```
az keyvault certificate create \
  --vault-name azuremol \
  --name molcert \
  --policy "$(az keyvault certificate get-default-policy)"
```

2. Чтобы увидеть этот сертификат в действии, создайте другую виртуальную машину, такую как `molwinvm`. На этот раз создайте ВМ с Windows Server 2019, чтобы попробовать другую операционную систему, и увидеть, что эти функции Key Vault не зависят от какой-то конкретной ОС! Укажите собственные имя пользователя и пароль администратора:

```
az vm create \
  --resource-group azuremolchapter15 \
  --name molwinvm \
  --image win2019datacenter \
  --admin-username azuremol \
  --admin-password P@ssw0rd1234
```

3 Можно автоматически добавить сертификат в виртуальную машину прямо из интерфейса командной Azure CLI. Этот подход не полагается на управляемое удостоверение — Azure внедряет сертификат с помощью агента ВМ Windows Azure.

Добавьте свой сертификат, например `molcert`, в созданную на шаге 2 ВМ, например `molwinvm`:

```
az vm secret add \
  --resource-group azuremolchapter15 \
  --name molwinvm \
  --keyvault azuremol \
  --certificate molcert
```

4 Подключитесь к ВМ и убедитесь, что сертификат интегрирован правильно. Чтобы подключиться к виртуальной машине, сначала получите ее публичный IP-адрес:

```
az vm show \
  --resource-group azuremolchapter15 \
  --name molwinvm \
  --show-details \
  --query [publicIps] \
  --output tsv
```

Для подключения к виртуальной машине используйте локальный клиент подключения к удаленному рабочему столу Microsoft на своем компьютере. Используйте учетные данные для подключения к виртуальной машине localhost\azuremol, а не учетные данные своего локального компьютера по умолчанию, которые ваш клиент удаленного рабочего стола может попытаться использовать, как показано на рисунке 15.9.

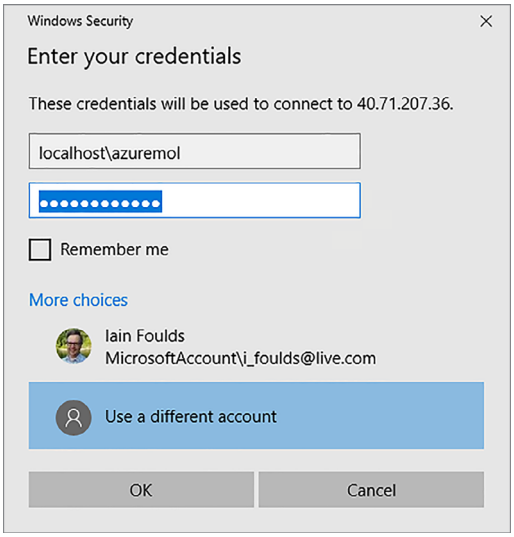


Рис. 15.9. Клиент удаленного рабочего стола может попытаться использовать ваши учетные данные локального компьютера по умолчанию. Вместо этого выберите «Использовать другую учетную запись» и укажите учетные данные localhost\azuremol, заданные вами при создании виртуальной машины.

5 После входа в систему нажмите кнопку «Пуск» в Windows и введите `mmc`, а затем откройте консоль управления Microsoft.

6 Выберите «Файл» > «Добавить/Удалить оснастку» и выберите параметр для добавления оснастки «Сертификаты».

7 Добавьте сертификаты для учетной записи «Компьютер», нажмите кнопку «Далее», а затем — «Готово».

8 Нажмите кнопку «ОК», чтобы закрыть окно «Добавление/Удаление оснастки».

9 Разверните папку «Сертификаты (локальный компьютер)» > «Личные» > «Сертификаты». В списке содержится сертификат из Azure Key Vault, внедренный в виртуальную машину, например CLIGetDefaultPolicy, как показано на рисунке 15.10.

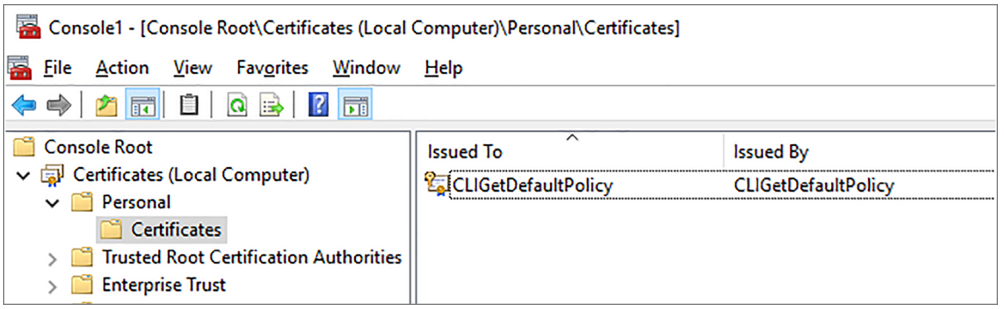


Рис. 15.10. На консоли управления Microsoft добавьте оснастку «Сертификаты» на локальный компьютер. Разверните хранилище «Личные» > «Сертификаты», чтобы просмотреть установленные сертификаты. В списке содержится сертификат, внедренный из Key Vault.

Это все, что от вас требуется! Создайте сертификат в хранилище ключей, а затем добавьте сертификат в виртуальную машину. Сертификат помещается в локальное хранилище сертификатов компьютера, что позволяет любому сервису или приложению получить к нему доступ. На виртуальной машине с Windows сертификаты хранятся в локальном кэше сертификатов, как показано в этом упражнении. На виртуальных машинах Linux PRV- и CRT-файлы для закрытых и открытых частей сертификата хранятся в папке /var/lib/waagent/. Затем можно перемещать сертификаты в любое расположение для своего приложения или сервиса.

Сертификаты можно использовать для аутентификации между клиентами и серверами или между компонентами приложений и сервисами. Типичный пример заключается в использовании SSL-сертификата веб-сервером — именно это вы будете наблюдать, выполняя практическое упражнение в конце главы.

15.5 Практическое упражнение: настройка безопасного веб-сервера

В последнем упражнении вы внедряли самозаверяющий сертификат из хранилища ключей Azure в виртуальную машину с Windows. Для этого практического упражнения установите и настройте веб-сервер IIS для использования сертификата, как показано далее:

1 Откройте PowerShell на виртуальной машине Windows и установите веб-сервер IIS:

```
Add-WindowsFeature Web-Server -IncludeManagementTools
```

2 Откройте диспетчер IIS. Это можно сделать в меню «Сервис» в диспетчере сервера.

3 В поле «Веб-сайт по умолчанию» выберите «Изменить привязки».

4 Добавьте привязку HTTPS на все неназначенные IP-адреса порта 443.

5 Выберите самозаверяющий сертификат, который вы создали и внедрили из Key Vault. Как правило, он называется CLIGetDefaultPolicy.

6 Откройте веб-браузер на виртуальной машине и введите `https://localhost`. Вы создали самозаверяющий сертификат в сервисе Key Vault, поэтому веб-браузер ему не доверяет.

7 Примите предупреждение, чтобы продолжить, и убедитесь, что привязка HTTPS работает.

8 В Azure Cloud Shell или на портале Azure создайте правило NSG для виртуальной машины в TCP-порту 443. Введите `https://yourpublicipaddress` в веб-браузере на локальном компьютере. Именно это будут видеть ваши пользователи, с предупреждением об использовании недоверенного самозаверяющего сертификата. В большинстве случаев помните о необходимости использовать доверенный внутренний или сторонний центр сертификации для создания доверенных сертификатов и хранить их в хранилище ключей.

16 Центр безопасности Azure и обновления

Было бы замечательно, если бы с помощью Azure можно было осуществлять мониторинг всех базовых ресурсов приложений и получать оповещения о проблемах безопасности, не так ли? А что, если политики безопасности вашей организации уже определены (если нет, то прямо сейчас поставьте себе напоминание создать такие политики!)? Как обеспечить соблюдение нормативных требований в своих развертываниях Azure? Если вы когда-либо проходили аудит ИТ-безопасности, вы знаете, как интересно изучать список ошибок конфигурации в вашей среде, а особенно нарушений базовых принципов безопасности, которых можно избежать!

Центр безопасности Azure представляет собой централизованное расположение оповещений и рекомендаций по безопасности, которые группируются и предоставляются вам для изучения. Можно определить собственные политики безопасности, а затем предоставить Azure возможность отслеживать состояние ваших ресурсов и контролировать их соответствие.

В этой главе мы рассмотрим, как центр безопасности может предупредить вас о проблемах и порекомендовать меры их устранения, как использовать возможность своевременного доступа к виртуальной машине для контроля и аудита удаленных подключений, а также как автоматически поддерживать виртуальные машины в актуальном состоянии с помощью системы управления обновлениями и новейших исправлений безопасности.

16.1 Центр безопасности Azure

В этой книге мы много раз обсуждали вопросы безопасности, например создание и настройку групп безопасности сети (NSG) для ограничения доступа к виртуальным машинам, а также ограничение трафика на учетные записи хранения Azure только зашифрованным трафиком. Что касается ваших собственных развертываний, а не упражнений в этой книге: знаете ли вы, с чего начать и как проверить, что соблюдены все рекомендации по безопасности? В этом вам поможет центр безопасности Azure — он проверит, о каких частях своей среды вы могли забыть.

Центр безопасности Azure сканирует ваши ресурсы, рекомендует исправления и помогает устранять проблемы безопасности, как показано на рисунке 16.1. Если вы работаете всего с парой тестовых виртуальных машин и одной виртуальной средой в своей подписке Azure, отслеживать необходимые ограничения безопасности может быть не так сложно. Однако по мере расширения вашей системы до десятков, сотен и даже тысяч виртуальных машин, следить за тем, какие конфигурации безопасности применить к той или иной виртуальной машине, может быть очень сложно.

Рис. 16.1. Центр безопасности Azure отслеживает ваши ресурсы Azure и использует определенные политики безопасности, чтобы уведомлять вас о потенциальных угрозах и уязвимостях. Предоставляются рекомендации и шаги по устранению проблем. Можно использовать своевременный доступ к виртуальным машинам, отслеживать и применять обновления безопасности, а также контролировать внесенные в белый список приложения, которые могут выполняться на виртуальных машинах.

Центр безопасности также может уведомлять вас об общепринятых передовых практиках, например если на виртуальной машине не включена диагностика. Помните, в главе 12 мы рассматривали мониторинг и диагностику виртуальных машин? Необходимо установить и настроить агент диагностики *до того,* как возникнет проблема. Если вы подозреваете нарушение безопасности, возможно, вы не сможете осуществить доступ к виртуальной машине и изучить журналы. Однако если вы настроили расширение диагностики для потоковой передачи журналов в хранилище Azure, вы сможете изучить произошедшее и, при благоприятном стечении обстоятельств, отследить источник и степень проблемы.

Попробуйте сейчас

Чтобы начать работу с центром безопасности Azure, выполните следующие действия:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Создайте группу ресурсов, введите имя, например `azuremolchapter16`, и расположение, например `eastus`:

```
az group create --name azuremolchapter16 --location eastus
```

3 Создайте базовую виртуальную машину с Linux, чтобы центру безопасности было что отслеживать и для чего предоставлять рекомендации:

```
az vm create \
  --resource-group azuremolchapter16 \
  --name azuremol \
  --image ubuntults \
  --admin-username azuremol \
  --generate-ssh-keys
```

4 После развертывания виртуальной машины закройте Cloud Shell.

5 На портале Azure выберите из списка сервисов слева «Центр безопасности». При первом открытии панели мониторинга потребуется несколько секунд, чтобы подготовить все доступные компоненты; см. рисунок 16.2.

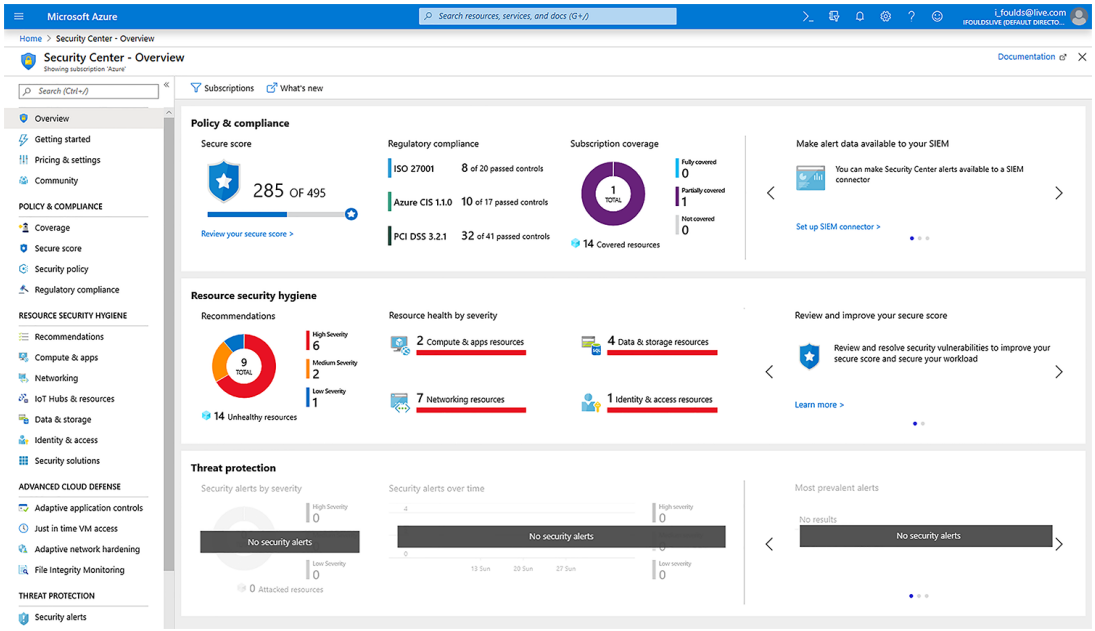


Рис. 16.2. В окне «Обзор центра безопасности Azure» приводится список рекомендаций, оповещений и событий. Можно выбрать базовый тип ресурса, например «Вычисления» или «Сети», чтобы просмотреть список элементов безопасности, которые относятся к этим ресурсам.

Центр безопасности контролирует развертывание таких ресурсов, как виртуальные машины, правила NSG и системы хранения данных. Встроенные базовые показатели безопасности используются для выявления проблем и предоставления рекомендаций. Например, только что развернутая вами виртуальная машина генерирует несколько предупреждений, как показано на рисунке 16.3. Можно и следует внедрить собственные политики безопасности, которые указывали бы Azure, как ограничить доступ и что нужно сделать для выполнения требований вашего бизнеса. По мере того как вы создаете или обновляете ресурсы, Azure

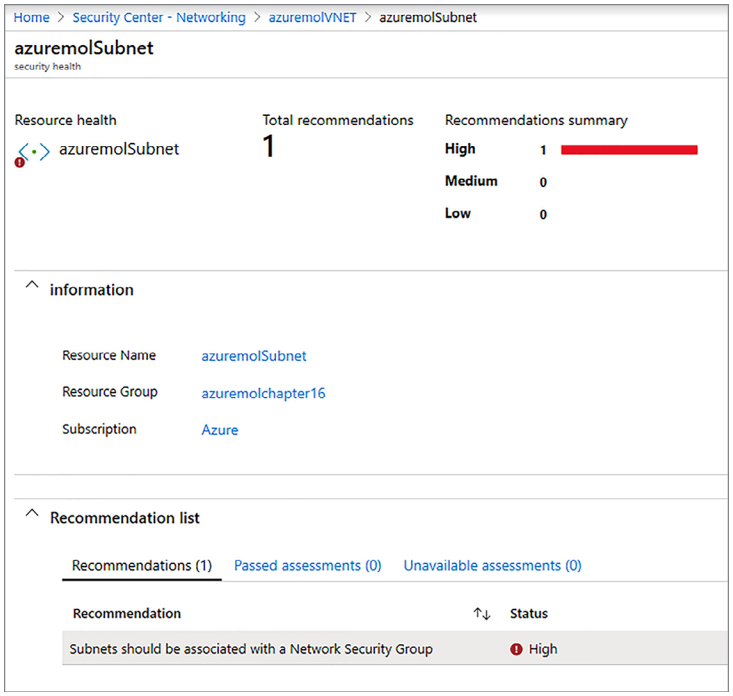


Рис. 16.3. Виртуальная сеть для ВМ уже создает предупреждения системы безопасности. В этом примере она предупреждает о том, что группа безопасности сети должна быть связана с подсетью.

непрерывно отслеживает происходящее на предмет отклонения от этих политик и уведомляет вас о том, какие действия нужно выполнить для устранения проблем безопасности. В этой главе используются политики безопасности Azure по умолчанию, но подумайте о конкретных конфигурациях безопасности, которые вы, возможно, захотите применить к своим виртуальным машинам, и как их можно определить в ваших настраиваемых политиках.

6 Выберите «Вычислительные ресурсы и приложения» в меню слева в окне центра безопасности. Затем выберите «Виртуальные машины и компьютеры».

7 Выберите виртуальную машину, созданную на шаге 3. Несмотря на то что вы только что создали эту виртуальную машину и использовали значения по умолчанию из интерфейса командной строки Azure, здесь показаны некоторые предупреждения безопасности.

Изучите некоторые из рекомендаций. При выборе некоторые рекомендации предоставляют больше информации, а другие — инструкции по устранению. Это не жесткие правила, а рекомендации. Некоторые из них могут не быть не применимы к вашей среде. Но это хорошая отправная точка, позволяющая понять, что делать для защиты ресурсов при их создании в Azure.

16.2 Ограниченный по времени доступ

В разделе 16.1. вы узнали, как центр безопасности предлагает ограничить область удаленного входящего подключения. Для ограничения трафика можно указать диапазон IP-адресов, однако в идеале открывать входящее подключение только в случае крайней необходимости. Так виртуальная машина будет полностью закрыта для удаленных подключений и доступна только в течение короткого периода, когда это

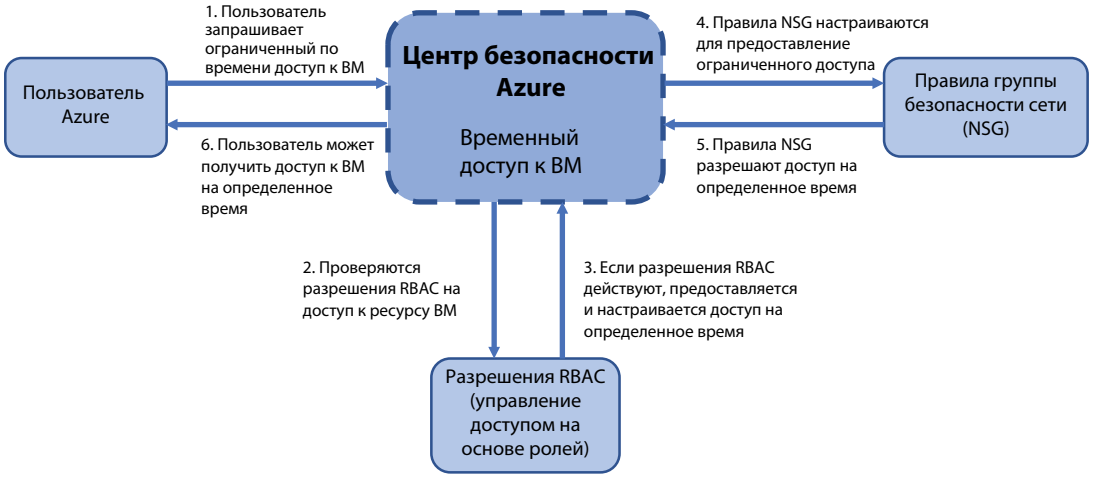


Рис. 6.4. При использовании ограниченного по времени доступа к виртуальной машине правила NSG настраиваются так, чтобы отклонять удаленные подключения к виртуальной машине. Разрешения RBAC используются для проверки разрешений, когда пользователь запрашивает доступ к виртуальной машине. Эти запросы проверяются, и если запрос удовлетворен, правила NSG обновляются таким образом, чтобы разрешить трафик из заданного диапазона IP-адресов в течение указанного периода. Виртуальная машина доступна пользователю только в это время. По истечении этого периода правила NSG автоматически возвращаются в состояние запрета.

нужно. И да, краткий период подключения также необходимо ограничить определенным диапазоном IP-адресов! И именно в этом случае оказывается полезным ограниченный по времени доступ к виртуальной машине, как показано на рисунке 16.4.

При использовании ограниченного по времени доступа центр безопасности динамически корректирует ограничения доступа к виртуальной машине. Если эта функция включена, создаются правила NSG, которые отклоняют весь трафик удаленных подключений. Затем пользователь может запросить доступ к виртуальной машине только по мере надобности. В сочетании с управлением доступом на основе ролей (см. главу 6) центр безопасности при запросе подключения определяет, имеет ли пользователь права доступа к виртуальной машине. Если у пользователя есть разрешения, центр безопасности обновляет соответствующие правила NSG, чтобы разрешить входящий трафик. Эти правила применяются только в определенный период. Как только этот период истекает, правила отменяются и виртуальная машина снова становится закрытой для удаленных подключений. Если есть активное подключение к ВМ, оно не будет автоматически прервано после истечения периода. Вы можете завершить обслуживание или устранение неполадок и отключиться, когда будете готовы, но не сможете установить новое соединение, если не запросите JIT-доступ снова.

Использование нескольких компонентов

Мы не рассматривали брандмауэр Azure, но это ресурс виртуальной сети, который больше похож на локальный физический брандмауэр, чем на группы безопасности сети. Если вам нужно больше гибкости и контроля над трафиком, брандмауэр Azure будет прекрасным вариантом, хотя за него нужно платить.

Не углубляясь в брандмауэр Azure, я хочу отметить, что центр безопасности Azure также интегрируется с ним, чтобы открывать и закрывать необходимые правила. Если вы

применяете брандмауэр Azure, а не только группы безопасности сети, для защиты трафика ВМ в виртуальных сетях, вы по-прежнему можете использовать автоматизированные правила управления доступом к ВМ на основе JIT.

Дополнительные сведения о брандмауэре Azure см. по адресу https://docs.microsoft.com/azure/firewall/overview.

Когда следует использовать ограничение доступа по времени в нашем вымышленном магазине пиццы? Подумайте о виртуальных машинах, на которых будут выполняться ваши веб-приложения, системы обработки заказов или приложения бизнес-логики. Вы бы хотели, чтобы они были подключены к Интернету и постоянно доступны любым пользователям? Надеюсь, что нет! Существуют веские причины для защиты удаленного доступа с помощью протоколов SSH и RDP, однако период доступности ВМ все равно нужно свести к минимуму. Даже если существуют правила NSG, ограничивающие доступ к определенным диапазонам IP-адресов, ограничение доступа по времени добавляет еще один уровень защиты доступа для пользователей Azure, а затем создает упрощенный аудиторский след, о котором центр безопасности может предоставлять отчеты.

Попробуйте сейчас

Чтобы включить ограниченный по времени доступ к виртуальным машинам, выполните следующие действия:

1. Откройте портал Azure и выберите «Центр безопасности» в меню слева.
2. В разделе «Расширенная защита облака» выберите «JIT-доступ к виртуальным машинам».
3. В ответ на соответствующий запрос выберите «Опробовать JIT-доступ к виртуальной машине» или «Обновить до стандартного уровня центра безопасности». Эта бесплатная пробная версия доступна 60 дней и не продлевается автоматически. Она пересекается с вашей бесплатной учетной записью Azure, поэтому использование этой возможности для вас бесплатно. Выберите вариант «Применить стандартный план», а затем подождите несколько минут, пока он включится. После этого вам, возможно, потребуется закрыть и открыть повторно портал Azure, чтобы выполнить следующие действия.
4. Снова выберите в окне центра безопасности вариант «JIT-доступ к виртуальной машине». После включения учетной записи стандартного уровня можно просмотреть список виртуальных машин, которые требуется использовать.
5. Выберите ВМ и нажмите кнопку «Запросить доступ», как показано на рисунке 16.5.

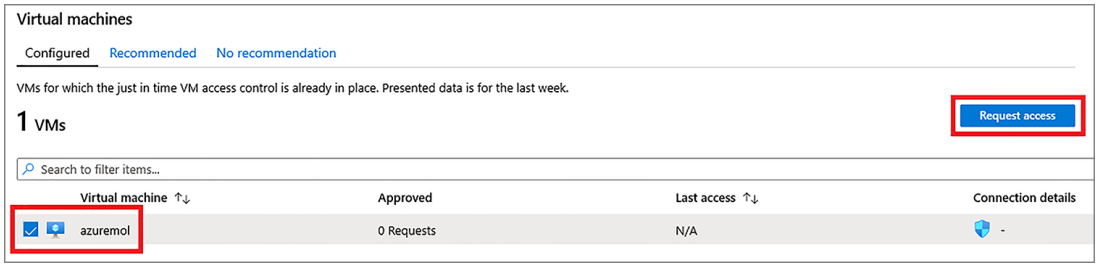


Рис. 16.5. Выберите виртуальную машину из числа рекомендованных, а затем включите JIT-доступ на ней. В настоящее время состояние показывает, что эта виртуальная машина открыта для любого удаленного доступа, ввиду чего серьезность проблемы безопасности отмечена как Высокая (High).

По умолчанию JIT определяет правила, которые могут открывать порты для SSH (порт 22), RDP (порт 3389) и удаленного взаимодействия PowerShell (порты 5985 и 5986) на 3 часа.

6 Для этого упражнения включите SSH с собственного IP-адреса. В производственной среде рекомендуется ввести обоснование, чтобы отслеживать причину запроса доступа. Оставьте все значения по умолчанию и нажмите «Открыть порты», как показано на рисунке 16.6.

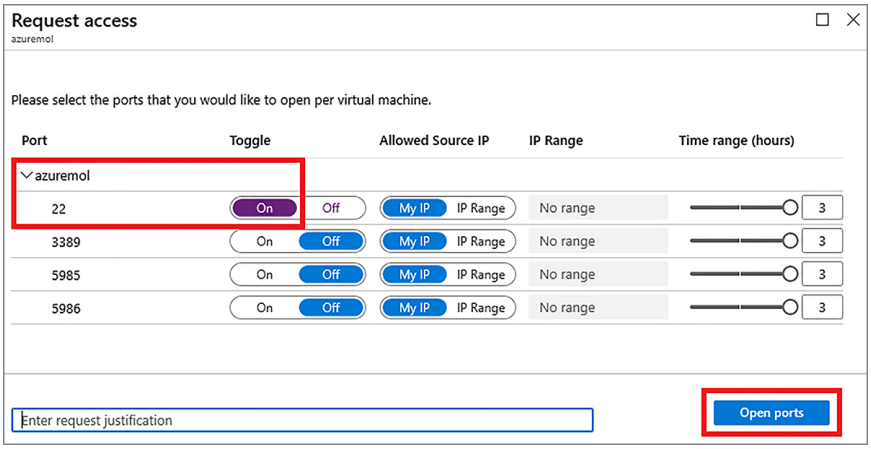


Рис. 16.6. При включении JIT можно изменить правила по умолчанию, которые следует разрешить, разрешенные исходные IP-адреса и максимальное время запроса в часах. Эти правила JIT обеспечивают точный контроль над разрешенными действиями и позволяют обеспечить минимально необходимые возможности подключения.

7 С включенными JIT-правилами найдите свою группу ресурсов и выберите свою виртуальную машину.

8 Выберите «Сеть» для просмотра назначенной виртуальной машине конфигурации виртуальной сети. Отобразится список назначенных правил NSG, как показано на рисунке 16.7.

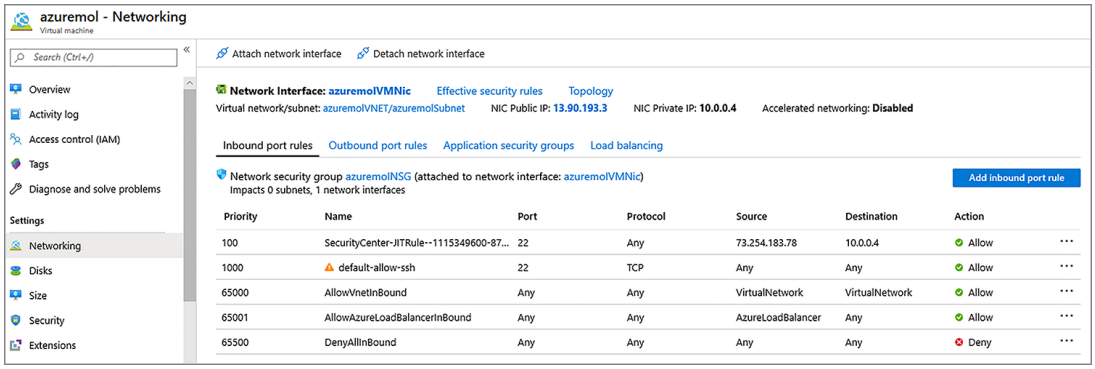


Рис. 16.7. Правила JIT создаются с наименьшим приоритетом. Эти приоритеты гарантируют, что правила JIT имеют приоритет над любыми правилами, которые впоследствии применяются на уровне подсети.

Правила JIT отображаются в верхней части списка, так как они имеют наименьший приоритет. Трафик разрешается на IP-адрес ВМ, но только с вашего IP-адреса. Это обусловлено JIT-доступом. Может показаться странным, что правило default-allow-ssh до сих пор существует и разрешает весь трафик. Вспомните главу 5, где мы обсуждали группы безопасности сети. Можете ли вы сказать, что происходит здесь?

JIT применяется только к виртуальной машине. Параметр Destination (Целевое расположение) в правиле JIT отображает IP-адрес виртуальной машины. В примере на рисунке 16.7 это 10.0.0.4. Трафик разрешен. Однако фактическое правило NSG применяется ко всей подсети. Правило default-allow-ssh применяется на уровне подсети t и разрешает трафик из любого источника и в любое место назначения.

Правила NSG обрабатываются в порядке приоритета, от низкого до высокого. Как обсуждалось в главе 5, действие «Запретить» вступает в силу всегда, независимо от дополнительных правил. Даже если изменить это правило default-allow-ssh и запретить трафик, правило JIT все равно разрешит доступ к конкретной ВМ и с определенного исходного IP-адреса.

Будьте осторожны с этим наслоением правил групп безопасности сети. В идеале следует удалить правило default-allow-ssh, а затем разрешать доступ только по мере необходимости с помощью JIT. При таком подходе SSH-подключается отклоняется окончательным правилом DenyAllInbound. Если вам необходимо подключиться к ВМ, используйте JIT для запроса доступа. При этом автоматически создается правило, разрешающее SSH-подключение для вашего IP-адреса в течение определенного периода времени.

Правило NSG удаляется автоматически после истечения указанного периода времени. По умолчанию правила JIT действуют 3 часа. Затем виртуальная машина возвращается в безопасное состояние и вам требуется снова запрашивать доступ к ней.

Этот JIT-процесс предназначен для управления тем, кто может запрашивать и получать доступ к виртуальной машине. Успешный запрос доступа к виртуальной машине не означает, что у пользователя будут разрешения на вход в систему на этой виртуальной машине. Всего-навсего в Azure обновляются установленные правила NSG. Центр безопасности и JIT не могут добавлять, удалять или обновлять учетные данные доступа на виртуальной машине.

Все запросы JIT также регистрируются. В центре безопасности выберите JIT-доступ к виртуальной машине, а затем выберите свое правило. Справа щелкните пункт меню «…» и затем выберите «Журнал действий». Этот журнал действий позволяет проверять, кто запрашивал доступ к ВМ, в случае проблемы.

JIT-доступ к виртуальной машине — один из используемых центром безопасности и Azure способов поддерживать безопасность ваших виртуальных машин. Управление доступом к виртуальным машинам — важная составляющая безопасности, но как насчет приложений, библиотек и сервисом, выполняемых на виртуальной машине? Необходимо убедиться, что на виртуальной машине своевременно устанавливаются все новейшие обновления безопасности.

16.3 Управление обновлениями Azure

Центр безопасности Azure, среди прочего, может предоставлять информацию о состоянии любых требуемых виртуальной машине обновлений ОС. В своем магазине пиццы следует попытаться установить новейшие исправления безопасности и исправления для приложений. Вы наверняка не захотите запускать какие-либо системы с известными уязвимостями или областями атак, поэтому автоматизация обновлений этих систем и возможность отслеживать их соответствие требованиям повышают уровень вашей безопасности. Работая с приложениями, в которых хранятся данные о клиентах и платежные сведения, не рекомендуется запускать

какие-либо системы без установленных актуальных обновлений. И не забудьте спланировать среду тестирования, которая позволяет безопасно применять исправления системы безопасности и проверять, что они не вызовут проблем, прежде чем внедрить их в производственные системы!

Функция управления обновлениями интегрирована в виртуальные машины Azure. Она позволяет искать, исправлять обновления ОС и составлять соответствующие отчеты. Преимущество этого решения в том, что оно работает как в Windows, так и в Linux, а также в разных дистрибутивах Linux, таких как Ubuntu, Red Hat и SUSE. На рисунке 16.8 показано, как система управления обновлениями отслеживает и может устанавливать необходимые обновления.

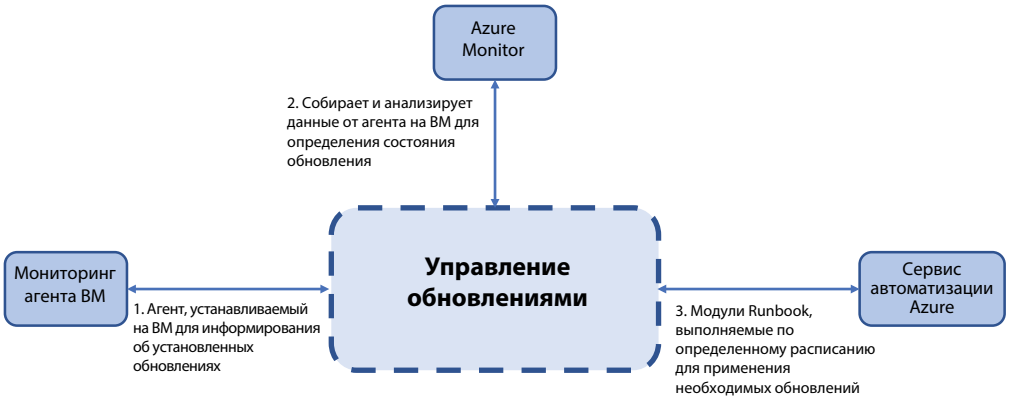


Рис. 16.8. Система управления обновлениями устанавливает агент виртуальной машины, который собирает информацию об установленных на каждой виртуальной машине обновлениях. Эти данные анализируются сервисом Azure Monitor, а соответствующие сведения возвращаются на платформу Azure. Затем с помощью модулей Runbook сервиса автоматизации Azure можно запланировать автоматическую установку необходимых обновлений по расписанию.

Подготовка виртуальной машины и отправка отчета о состоянии обновлений занимает несколько минут, поэтому давайте настроим вашу виртуальную машину и посмотрим, что происходит «за кадром».

Попробуйте сейчас

Чтобы настроить виртуальную машину для управления обновлениями, выполните следующие действия:

1. Откройте портал Azure и выберите «Группы ресурсов» в меню слева.
2. Выберите группу ресурсов, например `azuremolchapter16`, а затем выберите виртуальную машину, например `azuremol`.
3. В разделе «Операции» выберите «Управление обновлениями».
4. Оставьте значение по умолчанию для параметра «Расположение», создайте рабочую область Log Analytics и учетную запись сервиса автоматизации. В оставшейся части этого раздела мы рассмотрим эти компоненты подробнее.
5. Чтобы включить управление обновлениями для виртуальной машины, выберите «Включить».

Вы вернетесь в окно «Обзор управления обновлениями», однако требуется несколько минут, чтобы настроить виртуальную машину и отправить информацию о ее статусе. Пусть процесс выполняется, пока вы продолжаете чтение.

Давайте посмотрим, что обеспечивает работу функции «Управление изменениями».

16.3.1 Объединенные сервисы управления Azure

Если у вас есть опыт использования каких-либо локальных технологий Microsoft, возможно, вы уже сталкивались с набором инструментов System Center. System Center состоит из множества компонентов, таких как Configuration Manager, Operations Manager, Orchestrator и Data Protection Manager. Есть еще несколько компонентов, но именно вышеперечисленные составляющие обеспечивают следующие возможности:

- Определение конфигураций и требуемого состояния
- Установка приложений и обновлений
- Отчет о работоспособности и безопасности
- Автоматизация развертывания больших сервисов и приложений
- Резервное копирование и репликация данных

По мере того как все больше компаний за последние годы перешло на облачные вычисления, в традиционные локальные компоненты System Center были заменены на сервисы Azure, которые могут работать в гибридной среде. Мы уже рассмотрели 2 компонента в предыдущих главах, даже если вы не осознавали этого:

- *Сервис архивации Azure* позволяет выполнять резервное копирование виртуальных машин или отдельных файлов, определять политики хранения и восстанавливать данные.
- *Azure Site Recovery* позволяет реплицировать виртуальные машины в различные географические регионы в случае стихийного бедствия или длительного сбоя.

Сервис архивации Azure и Site Recovery уже помогли нам обеспечить безопасность данных в главе 13. Теперь в рамках управления обновлениями вы используете и другие сервисы:

- *Рабочие области Log Analytics* собирают информацию из различных источников и агентов и позволяет определить политики и запросы, чтобы уведомлять вас о наступлении различных условий. Эти запросы и оповещения могут помочь вам отследить состояние обновления виртуальной машины или уведомить вас о проблемах конфигурации или безопасности.
- *Azure Monitor* предоставляет сведения и отчеты на основе обработки, в рабочих областях Log Analytics. Azure Monitor реализует централизованный способ просмотра оповещений, запроса данных журналов и создания уведомлений для всех ресурсов Azure.
- *Сервис автоматизации* Azure позволяет создавать модули Runbook, выполняющие команды или целые сценарии. Модули Runbook могут представлять собой большие, сложные развертывания и могут вызывать множество других модулей Runbook. Сервис автоматизации Azure рассматривается подробно в главе 18.

Интеграция этих компонентов показана на рисунке 16.9.

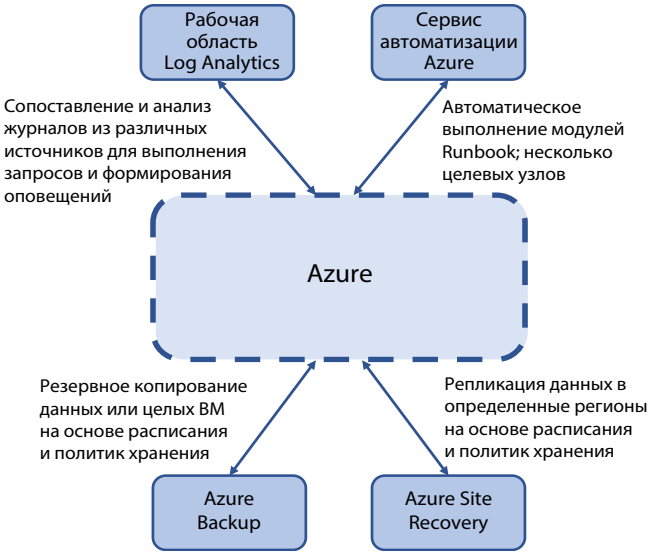


Рис. 16.9. Несколько сервисов Azure работают вместе, предоставляя функции управления и настройки для всей среды приложений. Сервисы, использующие эти компоненты, не ограничиваются виртуальными машинами или ресурсами Azure и при надлежащей настройке могут работать с другими облачными поставщиками или локальными системами.

Рабочие области Log Analytics и сервис автоматизации Azure — многофункциональные компоненты, которым можно было бы посвятить целые главы книги. Имея в управлении всего несколько виртуальных машин, вы вряд ли ощутите необходимость в центральном репозитории журнала для запросов и оповещений или способе автоматизировать развертывание и конфигурирование виртуальных машин. Если вы еще не начали составлять список компонентов Azure, которые вам хочется изучить поподробнее после прочтения этой книги, самое время такой список составлять — и не забудьте внести в него Log Analytics и сервис автоматизации Azure.

Следует понимать, что в Azure множество сервисов и компонентов могут взаимодействовать друг с другом и дополнять друг друга. Точно так же как виртуальные машины Azure и виртуальные сети Azure представляют собой отдельные, но взаимосвязанные сервисы, Log Analytics и сервис автоматизации Azure дополняют друг друга. Кроме того, работа этих компонентов зависит друг от друга. Сервис архивации Azure и расширение диагностики Azure — сами по себе отличные компоненты, но по-настоящему их возможности проявляются, если вы используете рабочие области Log Analytics и Azure Monitor для мониторинга состояния и сопоставления всех созданных событий и предупреждений. Надеюсь, вы уже начали выделять для себя некоторые взаимосвязанные компоненты и видите, как сервисы Azure тесно взаимосвязаны между собой. В заключительных главах книги мы рассматриваем варианты обеспечения безопасности и мониторинга, и наша цель — убедиться в работоспособности и стабильности приложений, которые вы выполняете в Azure.

Это называется «идентификация».

Продолжая разговор о сервисах, которые дополняют друг друга, не могу не вспомнить о большой (действительно *большой*!) составляющей Azure, о которой пока мы говорили лишь вскользь — это Azure Active Directory (Azure AD). Идентификация занимает центральное место в Azure, а AAD предоставляет некоторые функции безопасности, которые мы рассмотрели в главе 6 с моделью развертывания Azure Resource Manager. Возможность использовать систему управления доступом на основе ролей (RBAC) для ограничения прав определенных пользователей или групп выполнять те или иные действия с ресурсом, невозможна без функционирования централизованного решения идентификации. Azure AD обеспечивает даже возможность входа на портал Azure или в Azure CLI.

В этой книге не рассматривается Azure AD, поскольку эта технология предоставляет широкие возможности и сильно отличается от сервисов Azure IaaS и PaaS, таких как виртуальные машины, масштабируемые наборы и веб-приложения. Целевые аудитории этих двух тем могут пересекаться, однако большинство разработчиков, изучая Azure AD, преследуют иные цели, чем диспетчеры приложений или ИТ-специалисты, занимающиеся развертыванием инфраструктуры.

В зависимости от вашей учетной записи Azure ваши возможности в Azure AD могут быть ограничены. При регистрации для получения бесплатной пробной учетной записи Azure для вас создается экземпляр AAD по умолчанию. Вам принадлежит основная учетная запись в этом каталоге, и вы обладаете полными правами администратора. Если выполнить вход в Azure, используя учетную запись для своей компании или учебного заведения, велика вероятность, что ваши административные права будут ограничены или отсутствовать. Поэтому даже если мы выберем пару тем для рассмотрения, вы, возможно, не сможете напрямую выполнить ни одного упражнения. И я настоятельно не рекомендую вам выяснять, как все работает, в реальной среде Azure AD!

Однако Azure AD — это один из базовых сервисов Azure, который связывает воедино многие другие сервисы и компоненты. Облачные вычисления не упрощают вашу жизнь и не упорядочивают операционную среду каким-то «волшебным» образом — вам все равно нужно взаимодействовать с разными командами и заинтересованными лицами. Надеюсь, что изучив предыдущие главы этой книги, вы освоили базовые навыки работы с этими сервисами Azure и поняли, как создавать крупные приложения с избыточностью и эффективно общаться с коллегами, осознавая, какие задачи стоят перед другими командами вашей организации.

16.3.2 Просмотр и применение обновлений

Агенту виртуальной машины может потребоваться время для выполнения первого сканирования и подготовки отчета о состоянии примененных обновлений. Список установленных компонентов также должен содержать перекрестные ссылки на список доступных обновлений для заданной ОС и версии. Если обновления на вашей виртуальной машине не завершены и вы не получили отчета о состоянии, продолжайте читать материал и проверьте информацию снова через несколько минут. Как только процесс будет завершен, отобразятся обзорные сведения, как на рисунке 16.10. Будьте терпеливы: может потребоваться от 10 до 15 минут, чтобы готовность агента отображалась как «Готово» и вы могли запланировать установку обновлений.

Список необходимых обновлений — это прекрасно, но как их установить? И в этом вам поможет сервис автоматизации Azure! Когда вы включили управление обновлениями, было создано несколько модулей Runbook сервиса автоматизации Azure, которые автоматически обрабатывают процесс применения требуемых обновлений.

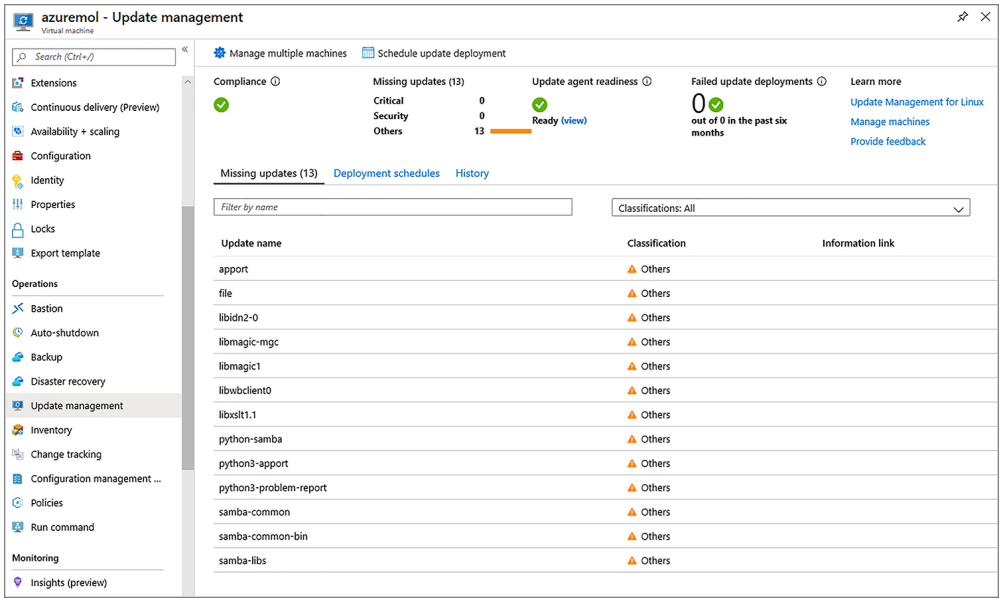


Рис. 16.10. После того как агент виртуальной машины просканировал ее на соответствие требованиям, отображается список доступных обновлений. В зависимости от ОС и версии система управления обновлениями, возможно, сможет вместе с рабочей областью Log Analytics и Azure Monitor классифицировать обновления по важности или предоставить ссылки на соответствующие страницы исправлений для этих обновлений.

Попробуйте сейчас

Если вам повезло (или не очень), виртуальная машина может сообщить, что никакие обновления не требуются. В Azure часто обновляются образы виртуальных машин, и если вы развертываете виртуальную машину вскоре после создания последнего образа, все необходимые обновления уже будут установлены. В этом случае прочитайте эти пошаговые инструкции, чтобы понять, что делать, когда виртуальной машине потребуется обновление.

Чтобы применить требуемые обновления к своей виртуальной машине, выполните следующие действия:

1. В разделе «Управление обновлениями» виртуальной машины выберите «Запланировать развертывание обновления».
2. Укажите имя развертывания обновления, например `azuremolupdates`, и изучите раздел «Классификации обновлений». Можно контролировать, какие наборы обновлений применяются к вашим виртуальным машинам. Пока же оставьте все параметры по умолчанию без изменений.
3. Раздел «Исключаемые обновления» позволяет задать определенные обновления, устанавливать которые не требуется. Если вы знаете, что приложению требуется определенная версия пакета или библиотеки, можно запретить установку обновленного пакета, чтобы не нарушать нужное вам состояние. Изучите доступные варианты, но в этом упражнении ничего менять не нужно.

4 Выберите «Параметры расписания» и укажите время применения обновлений, используя календарь и параметры времени. Время начала должно быть по меньшей мере на 5 минут позже текущего момента, чтобы платформа Azure могла обработать и запланировать ваш модуль Runbook в сервисе автоматизации Azure.

5 Когда все будет готово, нажмите кнопку «OK».

6 Если необходимо приостановить или завершить работу некоторых приложений и сервисов перед установкой обновлений и запустить их снова после завершения обновления, выберите «Предварительные скрипты + Последующие скрипты». Можно настроить отдельные задачи автоматизации для выполнения действий с виртуальными машинами до и после применения обновлений.

7 Окно обслуживания (в минутах) определяет, как долго может выполняться процесс обновления до того, как виртуальная машина должна будет снова вернуться в эксплуатацию. Окно обслуживания блокирует длительные обновления, из-за которых виртуальная машина может быть недоступна несколько часов подряд. Вы можете настроить короткие или длительные окна обслуживания (в зависимости от соглашений об уровне обслуживания) для приложений, выполняемых на этих виртуальных машинах, а также указать необходимые число и размер обновлений. Примите значение по умолчанию и нажмите кнопку «Создать».

8 В окне «Управление обновлениями» выберите «Расписания развертываний». Отобразится список обновлений, запланированных к установке в выбранные вами дату и время, как показано на рисунке 10.86.11.

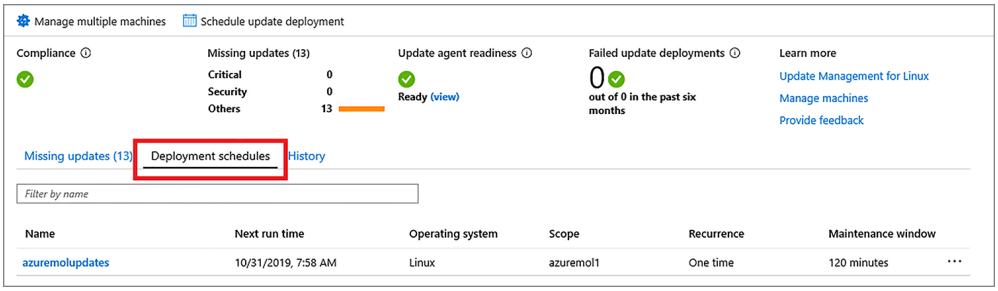


Рис. 16.11. Отображается список запланированных задач развертывания. При желании заданную задачу можно удалить; в противном случае обновления автоматически применяются в указанное время.

9 В верхней части окна «Управление обновлениями« выберите «Управление несколькими компьютерами». В окне отобразится учетная запись сервиса автоматизации Azure, которая была создана автоматически, когда вы включили управление обновлениями для этой виртуальной машины. Сейчас не следует слишком сильно беспокоиться о том, что делают модули Runbook. Вам не нужно ничего настраивать, а сервис автоматизации Azure рассматривается в главе 18.

Обратите внимание, что вы можете выбрать команду «Добавить виртуальную машину Azure» или «Добавить компьютер, не связанный с Azure», как показано на рисунке 16.12. Эта возможность еще раз обращает наше внимание на единый подход к управлению обновлениями в масштабах всей среды приложений, а не только на виртуальных машинах Azure.

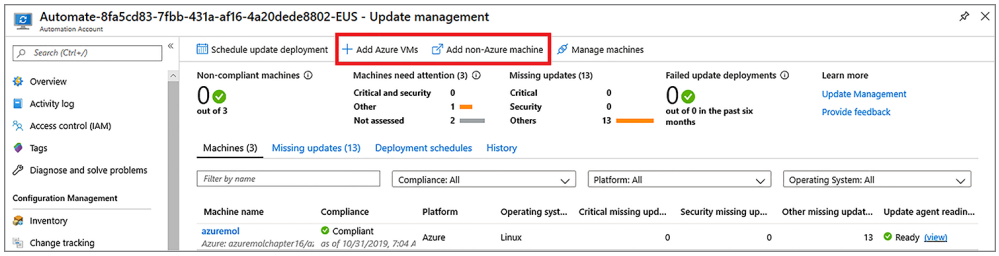


Рис. 16.12. В учетной записи сервиса автоматизации Azure можно управлять несколькими компьютерами, просматривать состояние или применять обновления. Виртуальные машины Azure и компьютеры, не связанные с Azure, можно отслеживать и контролировать с помощью одной учетной записи сервиса автоматизации Azure. Azure может интегрироваться с другими поставщиками, чтобы устанавливать агенты на компьютеры в гибридной среде. В результате такой интеграции создается единая панель мониторинга и платформа управления для работы с обновлениями.

10 Вернитесь в окно «Управление обновлениями» для вашей ВМ и откройте вкладку Журнал. Как только запустится развертывание обновления, отобразится его статус. Помните, что вы запланировали выполнение задания через несколько минут, поэтому оно не отображается сразу.

11 Выберите расписание, чтобы просмотреть статус состояние и результат, как показано на рисунке 16.13.

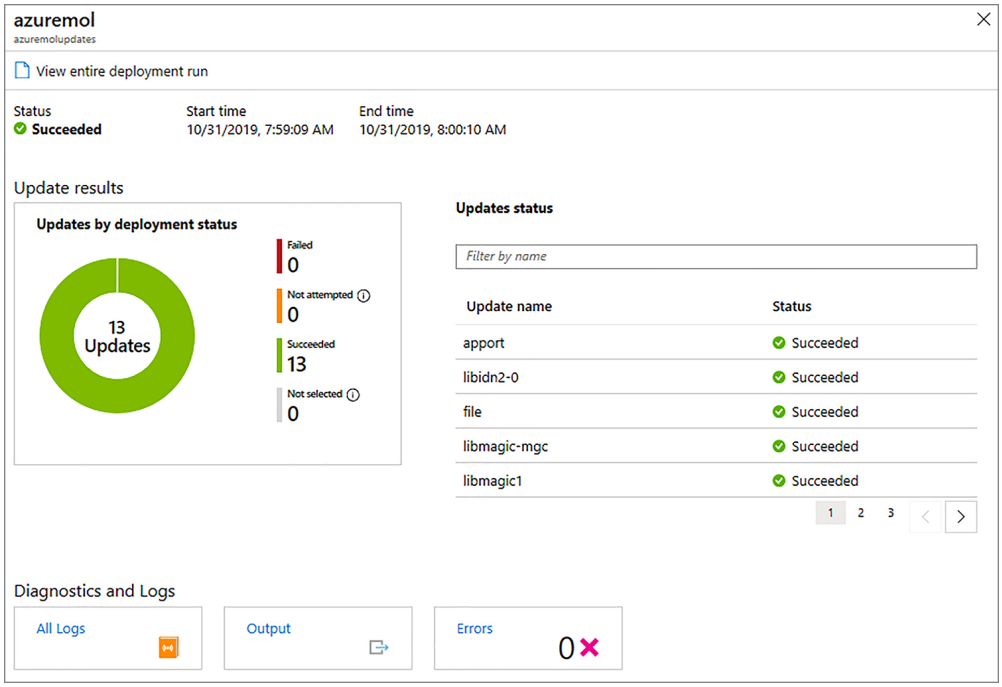


Рис. 16.13. Можно отслеживать состояние выполнения заданий сервиса автоматизации Azure на портале. Для просмотра и диагностики задач мощно щелкнуть задание, чтобы просмотреть созданные журналы и выходные данные.

12 По окончании развертывания обновления снова откройте свою группу ресурсов, выберите свою виртуальную машину и «Управление обновлениями». Обновление агента и отправка отчета в рабочей области Log Analytics о том, что обновления применены, могут занять несколько минут; после этого на панели мониторинга должна отобразиться информация о том, что виртуальная машина в актуальном состоянии и дополнительные обновления не требуются.

Это был краткий обзор центра безопасности и связанных компонентов, таких как JIT-доступ к виртуальной машине и управление обновлениями. Ваша цель — начать мыслить шире, нежели просто о том, как развернуть и запустить виртуальную машину или веб-приложение, и начать планировать управление приложениями более масштабно. Облачные вычисления не избавляют вас от необходимости реализовывать политики безопасности — напротив, потребность в обеспечении безопасности ресурсов возрастает! Компоненты Azure, такие как центр безопасности, помогут вам понять, что нужно сделать, а встроенные инструменты, такие как управление обновлениями и сервис автоматизации Azure — обеспечить безопасность вашей системы в любой момент.

16.4 Практическое упражнение: включение ограниченного по времени доступа и обновлений для виртуальной машины Windows

В этой главе рассматривается несколько компонентов, включение и подготовка отчета о статусе которых может занимать определенное время. Это практическое упражнение является необязательным, его основная цель — показать, что все эти функции работают независимо от используемой операционной системы. Если у вас нет времени или вы понимаете, как использовать эти функции на виртуальной машине Windows, упражнение можно пропустить. В противном случае попробуйте выполнить следующие задачи, чтобы попрактиковать использование центра безопасности и системы управления обновлениями. Практика — путь к совершенству, не так ли?

1 Создайте виртуальную машину Windows Server в той же группе ресурсов, которую вы использовали в предыдущих упражнениях, например в `azuremolchapter16`.
2 Просмотрите правила NSG для виртуальной машины или подсети и удалите все правила по умолчанию, разрешающие RDP на TCP-порту 3389.
3 Используйте локальный клиент подключения к удаленному рабочему столу, чтобы убедиться, что подкл ючения RDP заблокированы.
4 Запросите JIT-доступ, просмотрите правила NSG еще раз и подтвердите, что теперь можно установить RDP-подключение к вашей виртуальной машине.
5 Включите управление обновлениями на своей виртуальной машине Windows. На этот раз у вас должна быть возможность использовать существующие учетные записи для рабочей области Log Analytics и сервиса автоматизации Azure.
6 Предоставьте агенту мониторинга возможность подготовить для вас отчет о необходимых обновлениях, а затем запланируйте применение обновлений с помощью сервиса автоматизации Azure.

Часть 4

Самое интересное

А теперь самое интересное! В нескольких заключительных главах мы поговорим о технологиях будущего, которые можно использовать в Azure, таких как искусственный интеллект и машинное обучение, контейнеры, Kubernetes и Интернет вещей. Возможно, вы не пользуетесь этими сервисами сейчас, но учитывая актуальные тенденции в мире вычислений, скорее всего, в ближайшем будущем начнете. Это одни из самых интересных технологий, с которыми вы можете работать. Несмотря на то, что содержание книги довольно сжато, чтобы вы могли изучить эти темы во время обеденного перерыва, эта часть — отличный способ подвести итоги и показать вам, что вы можете создать в Azure.

17 Машинное обучение и искусственный интеллект

Надеюсь, то, что показывают в *Терминаторе* и *Матрице*, никогда не станет реальностью. В этих фильмах развитие искусственного интеллекта практически приводит к уничтожению человечества, когда машины берут вселенную под свой контроль. Одной из причин для беспокойства в современных вычислительных технологиях является то, что развитием искусственного интеллекта занимаются, в основном, крупные частные компании с минимальным или нулевым регулированием и централизованным надзором. И я не хочу сказать, что искусственный интеллект — это плохо! Цифровые помощники на смартфонах помогают в выполнении многих повседневных зада. Машинное обучение в приложениях для навигации предлагает альтернативные маршруты в зависимости от ситуации на дорогах и погоды. Системы регулирования температуры в жилых домах позволяют автоматически корректировать температуру в помещении с учетом температуры на улице, времени дня и времени года.

В заключительной части этой книги вы узнаете о сервисах Azure для машинного обучения и искусственного интеллекта. В одной главе. На своем обеденном перерыве. Будем реалистами: вы не станете экспертом в сфере машинного обучения или искусственного интеллекта в ближайшие 45 минут! Если вы пообедаете достаточно быстро, у вас будет время узнать о многочисленных сервисах Azure и о том, как интегрировать эти сервисы на базе машинного обучения и искусственного интеллекта в свои приложения. Для использования многих сервисов Azure на базе машинного обучения и искусственного интеллекта требуется опыт работы с алгоритмами данных, языками программирования, пакетной обработкой, а также знание языков программирования, так что не рассчитывайте стать экспертом в этой области за один час.

В этой главе нас ждет краткий обзор когнитивных сервисов Azure с функциями машинного обучения и искусственного интеллекта. Вы узнаете, как использовать эти сервисы для проведения базового машинного обучения на моделях данных, а затем вы сможете использовать сервис «Веб-приложения Azure» и Microsoft Bot Framework для запуска сервисов на базе ИИ, которые обеспечат работу бота в онлайн-магазине пиццы, помогающего клиентам разместить заказ.

17.1 Обзор и взаимосвязь искусственного интеллекта и машинного обучения

Держитесь крепче, потому что на следующих нескольких страницах я планирую как следует разогнаться. При разработке приложений в Azure искусственный интеллект и машинное обучение часто пересекаются. Давайте рассмотрим эти технологии по отдельности, а затем — как они взаимодействуют друг с другом.

17.1.1 Искусственный интеллект

ИИ позволяет компьютерам выполнять задачи с определенной степенью гибкости и осведомленности, а также корректировать свои решения на основе внешних факторов или без необходимости взаимодействия с человеком. Как правило, перед разработчиком не стоит цель создать полностью автономную систему, способную развиваться и «мыслить» самостоятельно, а, скорее, использовать набор моделей данных и алгоритмов в процессе принятия решений.

На ПК и смартфонах искусственный интеллект, как правило, представлен Siri, Кортаной и Google Ассистентом. Как показано на рисунке 17.1, эти ресурсы на базе искусственного интеллекта позволяют (часто с помощью голосовых команд) взаимодействовать, спрашивать направление движения, настраивать напоминания, выполнять поиск в Интернете и многое другое.

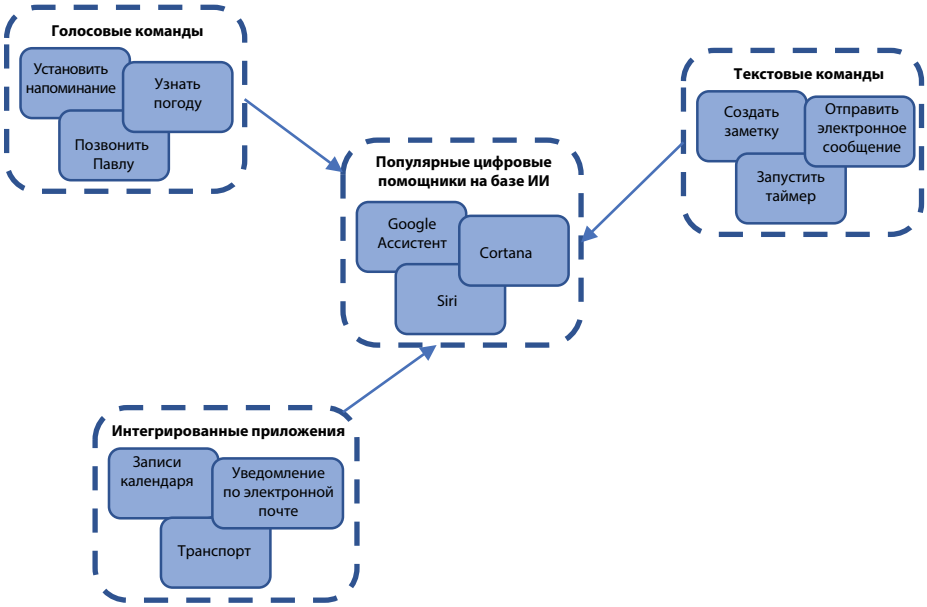


Рис. 17.1. Наиболее распространенной сферой применения искусственного интеллекта в повседневной жизни являются цифровые ассистенты: Кортана, Siri и Google Ассистент. Для взаимодействия с ними можно использовать голосовые или текстовые команды, а ассистенты могут отслеживать ваш ежедневный календарь и ситуацию на дорогах, чтобы предупреждать вас о возможных пробках.

Для работы таких цифровых ассистентов, как правило, не требуется то, что мы называем *интеллектом*, в большом объеме. Они слушают и реагируют на вводимые вами данные. Однако эти вводимые данные могут меняться и не всегда представлять собой конкретные команды. Подумайте о ситуации, когда цифровой помощник позволяет вам настроить напоминание. Можно использовать одну из следующих фраз:

- «В 5 часов напомни мне купить молоко».
- «Напомни мне купить молока по дороге домой».
- «Мне нужно купить молоко, когда я буду в магазине».

Если вы разработали традиционное приложение, вам нужно написать код, который мог бы обработать все возможные варианты того, как пользователь может предоставлять инструкции. Можно создать регулярные выражения, охватывающие несколько вариаций команд, но что произойдет, если пользователь произнесет незапрограммированную вами фразу? А что если пользователь допустит опечатку при вводе текста запроса, которую вы не предусмотрели? Подобные типы взаимодействия — отличный пример работы искусственного интеллекта. Как показано на рисунке 17.2, приложение программируется для обработки нескольких общих фраз, а затем может делать обоснованные предположения о том, чего хочет пользователь.

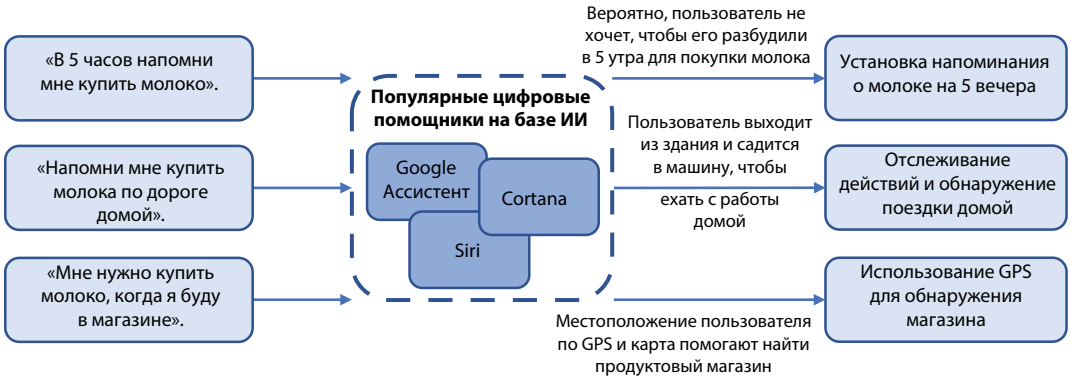


Рис. 17.2. Искусственный интеллект может принимать входные данные от пользователя и принимать решения, которые наиболее соответствуют ожидаемому действию. Все возможные ответы и деревья принятия решений не запрограммированы в искусственном интеллекте. Вместо этого искусственный интеллект использует модели и алгоритмы данных, чтобы применить контекст к входным данным пользователя, интерпретировать значение и вывести соответствующие выходные данные.

Пока это не истинный интеллект (такого нельзя сказать даже о сложных формах искусственного интеллекта); скорее, это обоснованное предположение на основе модели данных, с помощью которой этот искусственный интеллект был обучен. Эта модель данных может включать множество вариаций и фраз, а также со временем изучать новые значения. Как учится эта модель, и откуда поступают эти модели данных? И вот здесь на первый план выходит машинное обучение.

17.1.2 Машинное обучение

Большие данные стали настоящей сенсацией в мире вычислений в последние несколько лет. Концепция заключается в том, что компьютерные системы (особенно в облаке) — это отличный ресурс для обработки больших объемов данных. *Действительно* больших объемов данных. Задания обработки могут выполняться несколько минут или часов в зависимости от размера данных и требуемых вычислений. Они позволяют подготовиться и проанализировать большие объемы данных, чтобы определить конкретные закономерности и корреляции. Полученные сведения формируют модели данных, с помощью которых другие приложения или

Рис. 17.3. Большие объемы необработанных данных обрабатываются и подготавливаются к использованию. В зависимости от необработанных входных данных можно использовать разные техники подготовки и методики очистки данных. Алгоритмы машинного обучения затем применяются к подготовленным данным для создания подходящей модели данных, которая отражает оптимальную корреляцию между всеми точками данных. Можно создавать и со временем уточнять разные модели данных. Затем приложения могут использовать модели данных в работе с собственными входными данными, тем самым направляя процесс принятия решений и анализа закономерностей.

искусственный интеллект могут принимать решения. Как показано на рисунке 17.3, машинное обучение включает несколько этапов и как входные, так и выходные данные.

Вот как работает машинное обучение в самой простой форме:

1. В самом начале в качестве входных данных в систему поступают большие объемы необработанных данных.
2. Эти данные обрабатываются и приводятся в пригодный для использования формат, что позволяет сконцентрироваться на конкретных точках данных, необходимых для анализа.
3. К данным применяются алгоритмы машинного обучения. Именно на этом этапе выполняются самые интенсивные вычисления. Создаются алгоритмы для обнаружения и вычисления сходств и различий среди большого количества точек данных.
4. По результатам анализа алгоритмов создается модель данных, которая определяет закономерности в данных. Со временем эти модели данных могут уточняться, если части модели оказываются неверными или неполными при добавлении в систему новых реальных данных.
5. Приложения используют модели данных для обработки собственных наборов данных. Эти наборы данных обычно намного меньше, чем исходные данные, предоставляемые алгоритмам машинного обучения. Если модель данных действительна, то даже при небольшом объеме входных данных из приложения можно определить правильный результат или корреляцию.

Машинное обучение часто включает сложные алгоритмы, которые предназначены для обработки всех предоставленных точек данных. Hadoop и Apache Spark — это 2 распространенных стека приложений, которые используются для обработки больших данных. Azure HDInsight — это управляемый сервис, который позволяет анализировать большие наборы данных, обрабатываемые этими стеками приложений. Чтобы рассказать вам об анализе и алгоритмах чуть больше, добавлю, что специалисты по аналитике данных обычно используют для разработки нужных моделей язык программирования R. Не стоит слишком задумываться о том, что представляет собой Hadoop или R. Ключевой момент заключается в том, что Azure может использоваться для работы широко распространенных в отрасли инструментов машинного обучения.

17.1.3 Объединение искусственного интеллекта и машинного обучения

Одним из наиболее распространенных приложений на смартфоне является приложение для навигации, как показано на рисунке 17.4. Ваш провайдер, например Google, может отслеживать маршрут, по которому вы каждый день ездите на работу, в какое время вы обычно выходите из дома и сколько времени занимает у вас дорога на работу.

Данный пример с Google Maps показывает совместную работу искусственного интеллекта и машинного обучения. Искусственный интеллект используется, чтобы

узнать, когда создавать уведомление на основе данных, полученных в результате обработки модели данных машинного обучения. Другой пример совместного использования искусственного интеллекта и машинного обучения — уже упомянутая настройка напоминаний о покупке молока.

Рис. 17.4. Сервис Google Maps ежедневно получает от пользователей множество точек данных, фиксирующих сведения об их маршруте на работу и с работы. Эти данные можно подготовить и обработать вместе с прогнозом погоды и обновляемыми в реальном времени данными о погоде во время поездок на работу и с работы. К крупным наборам данных можно применить алгоритмы машинного обучения, а затем создать модель данных. В качестве мелкой выборки активных водителей можно отправить данные о текущих условиях поездки или погоде в сервис Google Maps, использовать модель данных для прогнозирования маршрута и отправить на смартфон оповещение о ситуации на дорогах с рекомендацией последовать домой другим маршрутом.

использования искусственного интеллекта и машинного обучения — уже упомянутая настройка напоминаний о покупке молока. Если искусственный интеллект был обучен с использованием моделей данных машинного обучения, ассистент бы знал, что вы покупаете молоко в продуктовом магазине, и не напоминал бы вам зайти за молоком в магазин инструментов. Модель данных машинного обучения также могла бы помочь ИИ понять, что с гораздо большей вероятностью вы хотите получить напоминание в пять часов вечера, а не в пять часов утра, поэтому ассистент не стал бы будить вас в пять утра напоминанием о покупке молока. Если ваш смартфон отслеживает, что в 17:00 вы садитесь в машину и едете с работы домой, машинное обучение создаст модель данных, прогнозирующую вашу поездку домой, а искусственный интеллект «поймет», что сейчас подходящее время напомнить вам о молоке.

Это простые, но очень показательные примеры, демонстрирующие как машинное обучение повышает качество искусственного интеллекта. Вы обучаете искусственный интеллект, предоставляя набор точек данных, которые обрабатываются системой машинного обучения для повышения точности и принятия решений.

17.1.4 Инструменты машинного обучения Azure для специалистов по аналитике данных

Я хочу вкратце рассказать о паре используемых в реальной жизни способов выполнения крупных вычислений и работы машинного обучения. Чтобы эта глава была понятна всем, в упражнениях используется платформа Microsoft Bot Framework для искусственного интеллекта и машинное обучение с интеллектуальным сервисом

распознавания речи (LUIS). Чтобы как следует погрузиться в мир машинного обучения, нужно сконцентрироваться на обработке данных и алгоритмах.

В Azure есть несколько потрясающих компонентов для анализа больших объемов данных. Во-первых, есть сервис машинного обучения Azure — веб-сервис, который позволяет визуально создавать эксперименты путем добавления наборов данных и моделей анализа. В этих экспериментах могут использоваться такие источники данных, как Hadoop и SQL, а также дополнительная поддержка программирования на языках R и Python. Можно перетаскивать источники данных, техники подготовки данных и алгоритмы машинного обучения. Можно скорректировать эти алгоритмы, а затем изучать и настраивать созданные в результате модели данных.

Цель сервиса машинного обучения Azure — сделать крупные вычислительные ресурсы Azure действительно доступными для большого числа пользователей. Основным преимуществом выполнения в Azure крупных вычислений данных на основе машинного обучения является то, что вы получаете доступ к большому объему вычислительных ресурсов и используете их только в течение времени, необходимого для выполнения вычислений. В традиционных средах эти дорогостоящие вычислительные ресурсы будут бездействовать в течение длительных периодов между заданиями по обработке данных.

Еще один эффективный ресурс, помогающий выполнять «серьезные» вычисления и модели машинного обучения в Azure — это виртуальные машины для обработки и анализа данных (DSVM). Такие виртуальные машины доступны как для Linux, так и для Windows. Они поставляются с множеством установленных стандартных приложений, такими как Jupyter Notebooks, Anaconda Python и R Server или SQL Server (см. рисунок 17.5).

Рис. 17.5. Виртуальные машины DSVM доступны для Windows и Linux. DSVM с Window Server 2016 поставляется с несколькими установленными приложениями для анализа и обработки данных, таких как R Server и Jupyter Notebooks. Виртуальные машины DSVM позволяют быстро приступить к обработке больших данных и созданию алгоритмов машинного обучения.

Нет необходимости устанавливать на локальном компьютере все инструменты и зависимости — можно создать DSVM с таким объемом ресурсов ЦП и памяти, который вам требуется для быстрой обработки данных, а затем, после того как задание по обработке данных будет выполнено и вы получите необходимые модели данных, — удалить виртуальную машину.

17.2 Azure Cognitive Services

Хорошо, а как насчет сервисов искусственного интеллекта, способных сделать ваши приложения «умнее»? В Azure ряд связанных сервисов составляет набор Cognitive Services. Эти сервисы охватывают несколько общих областей искусственного интеллекта, которые позволяют быстро интегрировать эти интеллектуальные ресурсы в ваши приложения. Они разделены на следующие области:

- Зрение
- Речь
- Язык
- Принятие решений
- Поиск

В семейство Cognitive Services входят больше 20 сервисов. Вот некоторые из них:

- *Зрение*, который включает в себя
  - Компьютерное зрение для анализа изображений, добавления титров и меток.
  - Распознавание лиц для анализа и обнаружения лиц на изображениях.
- *Речь*, который включает в себя
  - Speech Services для анализа и преобразования речи в текст и наоборот.
  - Распознавание говорящего для идентификации и проверки говорящего.
- *Язык*, который включает в себя
  - Распознавание речи (LUIS) для понимания пользователей и взаимодействия с ними. Мы рассмотрим сервис LUIS в практическом упражнении в конце этой главы.
  - Перевод текстов для анализа и исправления орфографических ошибок или перевода текста.
- *Принятие решений*, который включает в себя
  - Content Moderator для просмотра и модерации фото, видео и текста.
  - Персонализатор для анализа шаблонов и предоставления рекомендаций клиентам.
- *Поиск*, который включает в себя
  - Пользовательский поиск Bing для реализации поиска в пользовательских данных и приложениях.
  - Автозаполнение Bing для предоставления автоматических предложений по мере того, как пользователи вводя поисковые фразы и запросы.

Как видите, многие сервисы Azure объединяют технологии искусственного интеллекта и машинного обучения. В этой главе основное внимание уделяется языку, в частности службе LUIS. Как правило, она используется для создания интеллектуального бота для оказания помощи клиентам на вашем веб-сайте. Затем можно создать приложение, которое использует сервисы ИИ в Azure, может интерпретировать фразы и вопросы, а также давать соответствующие ответы, помогая клиенту оформить заказ или отправить запрос на обслуживание.

17.3 Создание интеллектуального бота, помогающего заказывать пиццу

Бот — это приложение, которое запрограммировано отвечать на задачи и вводимые пользователем данные. Кажется, речь идет об обычном приложении? Так и есть! Разница в том, как приложение-бот находит ответ.

Стандартный бот — не что иное как приложение, обеспечивающее определенную форму автоматизации. Когда пользователь отправляет сообщение, устанавливает тег в сообщении электронной почты или отправляет поисковый запрос, бот запускает запрограммированные задачи, которые выполняют определенное действие. По сути, здесь нет никакого искусственного обучения или машинного интеллекта. Приложение-бот просто реагирует на вводимые пользователем данные.

Наличие подходящей платформы позволяет расширить возможности бота, предоставить ему больше свободы и сделать его чуть «умнее». В начале обзора искусственного интеллекта я говорил о том, что стандартное приложение необходимо программировать, указывая все возможные варианты пользовательского ввода и соответствующие выходные данные. Однако если пользователь воспользуется другой вводной фразой или допустит опечатку, система не сможет правильно среагировать, так как не отличается гибкостью.

Microsoft предоставляет платформу Bot Framework, благодаря которой бот Azure может легко интегрировать наборы SDK Bot Builder и подключаться к Azure Cognitive Services. Даже с минимальным опытом написания кода вы сможете создавать интеллектуальных ботов, использующих возможности Azure для повышения качества обслуживания клиентов. Прошу вас об одном: не пытайтесь создать Скайнет, если не знаете, чем кончился *Терминатор*!

17.3.1 Создание бота веб-приложения на платформе Azure

Развернем бот и интегрируем некоторые сервисы на основе искусственного интеллекта и машинного обучения. Бот запускается в веб-приложении Azure, использует платформу Microsoft Bot Framework для подключения к LUIS и позволяет клиенту заказать пиццу. На рисунке 17.6 показано, что создается в результате этих действий и какие сервисы используются.

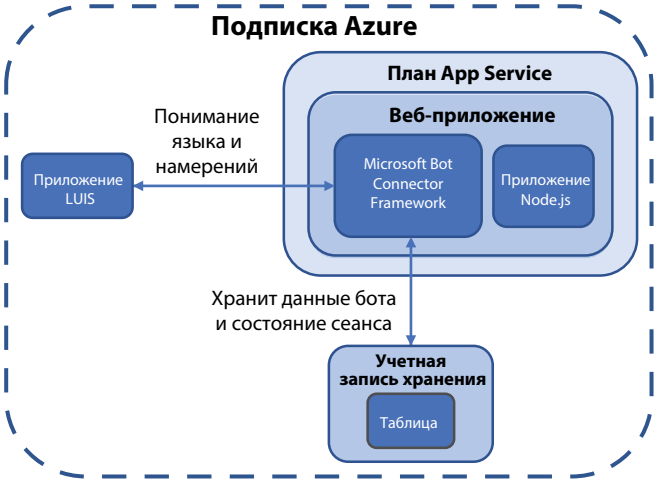


Рис. 17.6. В последующих упражнениях вы создадите бот веб-приложения, который интегрирует множество сервисов Azure на базе искусственного интеллекта и машинного обучения для взаимодействия с клиентом и оказания помощи в заказе пиццы.

Попробуйте сейчас

Чтобы создать бот веб-приложения Azure, выполните следующие действия:

1. Откройте портал Azure и в левом верхнем углу выберите «Создать ресурс».
2. Найдите и выберите «Бот веб-приложения», а затем нажмите кнопку «Создать».
3. Введите имя бота, например `azuremol`, а затем создайте новую группу ресурсов и введите имя, например `azuremolchapter17`.
4. Выберите наиболее подходящий регион и ценовой уровень F0. Вашему боту не придется обрабатывать много сообщений, поэтому бесплатного уровня (F0) будет достаточно.
5. Выберите шаблон бота и язык Node.js SDK.
6. Создайте базовый бот, так как мы приведем собственный пример кода приложения в следующем упражнении. На этом шаге создается приложение LUIS, которое можно использовать для обучения языку и машинного обучения.
7. Выберите подходящий регион для приложения LUIS и создайте новую учетную запись LUIS.
8. Введите имя учетной записи LUIS, например `azuremol`. Она будет оценивать настроение пользователя для нашего бота.
9. Выберите «План службы приложений» и создайте новый план. Введите имя, например `azuremol`, и выберите наиболее подходящий для вас регион.
10. Выключите App Insights, потому что ваш бот не будет использовать эту функцию. Как и в предыдущих главах, посвященных веб-приложениям, в производственной среде целесообразно использовать App Insights для получения сведений о производительности вашего приложения, наладив потоковую передачу данных и аналитики непосредственно из кода.
11. Оставьте включенным параметр для автоматического создания кода приложения и пароля Microsoft, примите условия соглашения и нажмите кнопку «Создать».

Создание бота веб-приложения и соответствующих компонентов занимает несколько минут. Многое происходит за кадром:

- Создается план служб приложений Azure.
- Развертывается веб-приложение и пример веб-приложения Node.js.
- Создается приложение LUIS, в веб-приложении настраиваются ключи подключения.
- В Microsoft Bot Connector создается бот, в веб-приложении настраиваются ключи подключения.

17.3.2 Язык и концепция распознавания в LUIS

Ранее мы рассматривали один из аспектов Azure Cognitive Service — язык. Это имеет смысл, потому что в той или иной форме язык часто используется для взаимодействия с искусственным интеллектом. Можно использовать LUIS для обработки сообщения или фразы пользователя и определения его намерений. Проанализировав намерение пользователя, приложение может дать ему правильный ответ. Давайте расширим возможности вашего бота с помощью сервиса LUIS.

Попробуйте сейчас

Чтобы создать приложение LUIS и обучить его с помощью машинного обучения, выполните следующие действия:

1. Откройте в браузере страницу www.luis.ai и войдите в систему, используя учетные данные вашей подписки Azure.
2. Нажмите «Перейти к моим приложениям» и выберите приложение, например `azuremol`. Скорее всего, в имени вашего приложения LUIS есть дополнительные символы (цифры), добавленные в него из указанного на портале Azure имени бота.

   Вам предложено несколько готовых намерений, но вы наверняка хотите записать в приложение LUIS примеры, которые больше подходят для вашего магазина пиццы.
3. Скачайте файл azuremol.json из GitHub по адресу https://github.com/fouldsy/ azure-mol-samples-2nd-ed/blob/master/17/luisapp/azuremol.json на ло кальный компьютер. Чтобы облегчить себе жизнь нажмите кнопку «Необработанные» на портале GitHub, чтобы просмотреть только содержимое файла.
4. Вернитесь в приложение LUIS, выберите «Управление приложением», а затем нажмите «Версии».
5. Выберите параметр для импорта версии, найдите и выберите скачанный файл azuremol.json, введите номер версии `1.0` и нажмите «Готово».
6. Вернитесь в раздел «Создание» в верхнем меню, чтобы просмотреть импортированные намерения из примера приложения. Выберите 1 или 2 намерения, например `greetings` или `orderFood`, и посмотрите на примеры фраз, которые клиент может использовать в общении с ботом.
7. Прежде чем вы сможете понаблюдать за приложением в действии, его необходимо обучить. Выберите «Обучить», а затем подождите несколько секунд, пока процесс не будет завершен. На рисунке 17.7 показано, как процессы машинного обучения используются для обучения вашего приложения LUIS.

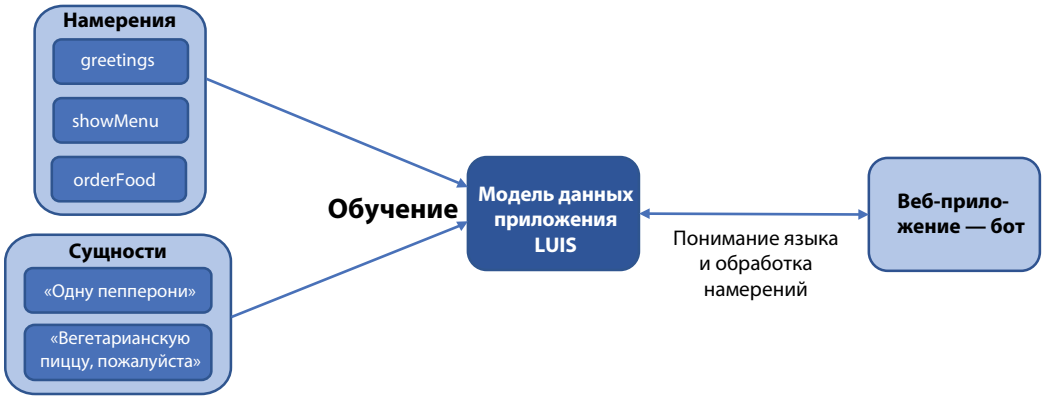


Рис. 17.7. При обучении приложения LUIS намерения и сущности вводятся и обрабатываются в системе с целью создания модели данных. Затем бот веб-приложения использует эту модель данных для распознавания речи и обработки намерений. Число вводимых намерений и сущностей для обработки невелико, поэтому модель данных неидеальна. В реальном мире вам предстоит иметь дело с гораздо большим количеством намерений и сущностей. Кроме того, вы будете непрерывно обучать, тестировать и уточнять свою модель данных для создания крупных наборов данных и, в конечном итоге, максимально точной модели для распознавания языка и обработки намерений.

В сложном реальном приложении процесс обучения может занять больше времени, поскольку все вводимые намерения и сущности обрабатываются алгоритмами машинного обучения с целью создания необходимой модели данных для вашего приложения, благодаря чему приложение сможет эффективно взаимодействовать с клиентом.

8 После обучения приложения LUIS выберите «Тест» и введите несколько приветствий, таких как *привет* и *здравствуйте*. Под каждым из сообщений отображается наиболее вероятное намерение, а также вероятность того, что введенное сообщение или высказывание соответствует намерению. Базовые приветствия должны точно соответствовать намерению приветствие.

9 Попробуйте ввести другое приветствие, например *(Добрый) день* или *(Добрый) вечер*. Состоящие из одного слова приветствия, указывающие на время суток, могут возвращать неверное наиболее вероятное намерение, например `orderStatus`. Пробуйте разные варианты фраз, пока не найдете что-то несоответствующее ожидаемому намерению, что будет свидетельствовать о том, что приложение LUIS не совсем хорошо понимает вас. Выберите одно из неверных сообщений, например *утро*, и нажмите «Проверить».

10 В меню «Проверка» выберите команду «Изменить» для неверного наиболее вероятного намерения. В раскрывающемся меню выберите приветствия или другое наиболее подходящее намерение для вашей неверной фразы.

11 Вы внесли изменение в приложение, поэтому выберите команду «Обучить приложение LUIS» еще раз. На рисунке 17.8 показано, как предоставить дополнительные входные данные для алгоритмов машинного обучения, чтобы затем обработать модель данных и уточнить распознавание речи и намерение.

12 В окне тестовых сообщений введите неверное сообщение еще раз, например *утро*. На сей раз наиболее вероятное намерение должно определиться правильно — приветствие.

13 Чтобы сделать обновленное приложение LUIS доступным вашему боту веб-приложения, выберите в верхнем меню вариант «Опубликовать». Оставьте все значения по умолчанию и выберите команду «Опубликовать в производственном слоте». Процесс публикации занимает несколько секунд.

Рис. 17.8. По мере того как вы вносите изменения в классификацию намерений сообщений и повторно обучаете приложение LUIS, модель данных уточняется, так как алгоритмам машинного обучения предоставляются дополнительные входные данные. Когда вы введете аналогичные приветствия в будущем, надеюсь, что усовершенствованная модель данных ответит вам подходящим способом.

Помните, ваш бот выполняется в веб-приложении, поэтому у него есть производственный и промежуточный слоты (помните, вы узнали об этом из 3 главы?). В реальной жизни нужно сначала опубликовать бот в промежуточном слоте, убедиться, что все работает должным образом, а затем опубликовать бот в производственном слоте. Функции PaaS, благодаря которым вы могли тестировать и перемещать интернет-код между жизненными циклами разработки и производственными жизненными циклами, также оказываются полезными в жизненном цикле вашего бота веб-приложения на основе LUIS.

Простой пример показывает, что система машинного обучения смогла распознать, что введенные вами данные *(Доброе) утро* — это приветствие, и понять, что аналогичные фразы, например *(Добрый) вечер*, также являются приветствиями. Машинное обучение проявляет себя лучше всего, когда есть возможность ввода крупного набора данных в модель данных, поэтому очень важно тщательно протестировать и обучить свое приложение. Искусственный интеллект (приложение LUIS в нашем случае) хорош ровно настолько, насколько хороши качество и размер данных, предоставляемых алгоритмам машинного обучения.

17.3.3 Создание и запуск бота веб-приложения с LUIS

Теперь у вас есть простейший бот веб-приложения в Azure и приложение LUIS, которое распознает речь и возвращает информацию о намерениях клиента. Чтобы интегрировать эти 2 компонента, вам необходимо изменить код вашего бота и обеспечить возможность использования LUIS. Пакеты SDK доступны для языков программирования C# и Node.js. Я считаю, что с Node.js несколько быстрее и проще понять, что происходит в коде, если все это для вас новый материал. Если вы знакомы с C#, вы вполне можете изучить набор SDK для языка C#, как только завершите эту главу. А сейчас давайте воспользуемся простейшим приложением Node.js из репозитория примеров GitHub, чтобы увидеть ваш бот в действии вместе с LUIS.

Попробуйте сейчас

Чтобы обновить бот веб-приложения обученным вами ботом LUIS, выполните следующие действия:

1. На портале Azure выберите «Группы ресурсов» в меню слева, выберите свою группу ресурсов, например `azuremolchapter17`, а затем выберите бот веб-приложения, например `azuremol`.

   Мы используем пример бота из репозитория образцов GitHub. Пример бота написан на Node.js, но, как и в предыдущих примера приложений, не волнуйтесь, если это не ваш язык.

2. Чтобы развернуть пример бота, откройте Cloud Shell. При необходимости скопируйте репозиторий примеров GitHub в Cloud Shell следующим образом:

```
git clone https://github.com/foulDsy/azure-mol-samples-2nd-ed.git
```

3. Перейдите в каталог для главы 17:

```
cd azure-mol-samples-2nd-ed/17/webappbot
```

4. Инициализируйте репозиторий Git и добавьте файлы бота:

```
git init && git add . && git commit -m "Pizza"
```

5. Чтобы загрузить пример бота, создайте подключение к веб-приложению. Следующая команда получает репозиторий веб-приложения и настраивает репозиторий Git с локальными примерами для подключения к нему. В предыдущих главах я просил вас заняться этой темой. Однако сейчас я надеюсь, что вы уже начали изучать возможности интерфейса командной строки Azure и поняли, что большую часть этой информации можно получить быстро.

```
git remote add webappbot \
  $(az webapp deployment source config-local-git \
  --resource-group azuremolchapter17 \
```

```
--name azuremol \
--output tsv)
```

6 Вставьте пример бота на Node.js в веб-приложение с помощью следующей команды:

```
git push webappbot master
```

7 При появлении запроса введите пароль для пользователя Git, которого вы создали и использовали в предыдущих главах (эта учетная запись была создана в главе 3).

Если вы не записали пароль Git на стикере

Если вы забыли пароль, его можно сбросить. Сначала получите имя пользователя локальной учетной записи развертывания Git:

```
az webapp deployment user show --query publishingUserName
```

Для сброса пароля введите имя учетной записи из предыдущей команды, а затем ответьте на запросы, чтобы задать новый пароль. В следующем примере сбрасывается пароль для учетной записи пользователя с именем azuremol:

```
az webapp deployment user set --user-name azuremol
```

Давайте посмотрим на рисунок 17.9, чтобы понять, что вы развернули. Теперь приложение LUIS обучено с помощью алгоритмов машинного обучения, и ваша модель данных готова к использованию в составе приложения Node.js, с помощью которого ваши клиенты будут взаимодействовать и заказывать пиццу.

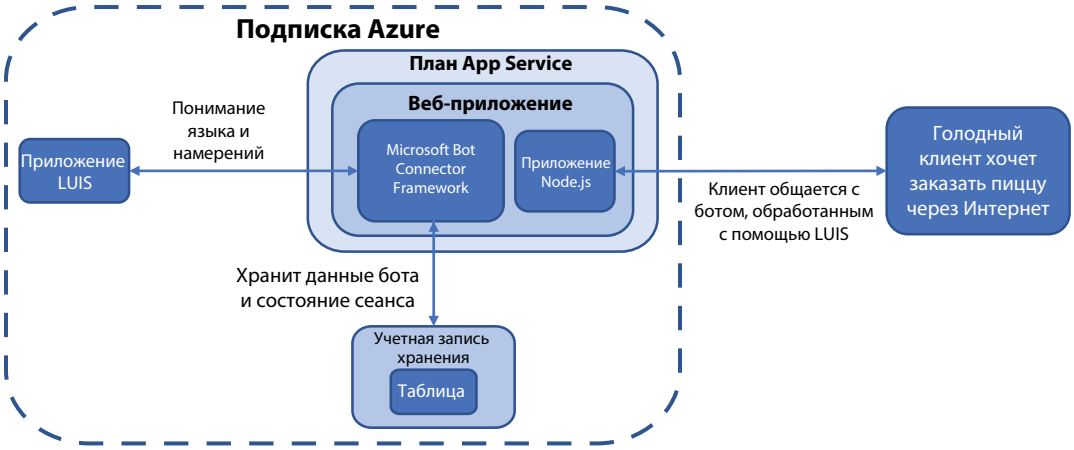


Рис. 17.9. Теперь клиент может связаться с вашим ботом в сети и попросить просмотреть меню или заказать пиццу. LUIS обеспечивает распознавание речи, благодаря чему бот может обрабатывать заказы и отправлять их в хранилище Azure для дополнительной обработки.

Вернитесь на портал Azure для бота веб-приложения и нажмите «Тестировать в веб-чате». Первое подключение к боту занимает несколько секунд, после чего вы сможете взаимодействовать, просматривать список пицц в меню, а также оформлять заказ, как показано на рисунке 17.10. Попробуйте сами!

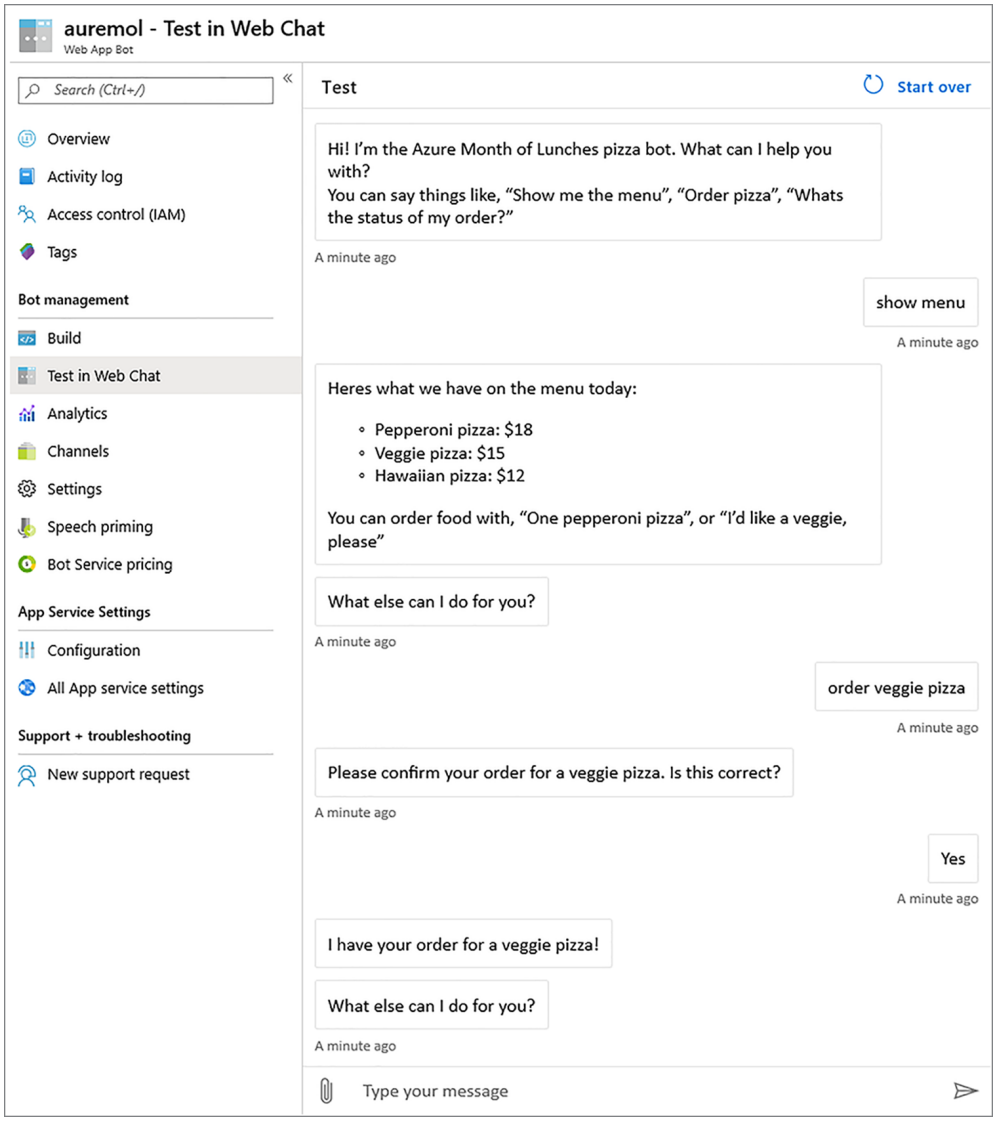


Рис. 17.10. Запустите свой бот веб-приложения, начните беседу и попробуйте заказать пиццу. В диалоговом окне из примера можно просматривать меню, заказывать пиццу и проверять состояние заказа. Это очень простое приложение, которое не создает заказы и не обновляет их статус, а только показывает, какую пиццу вы заказали, но я надеюсь, что теперь вы поняли, как быстро развернуть бот в Azure.

Надеюсь, из этих простых упражнений вы также получили представление о возможностях Azure в области искусственного интеллекта и машинного обучения. Бот веб-приложения с LUIS можно дополнить сервисами Azure Cognitive Service, такими как проверка орфография и переводчик. Эти сервисы позволяют интерпретировать слова и фразы даже в случае их неправильного написания пользователем. Кроме того, бот может общаться на нескольких языках. Кроме того,

можно использовать сервисы «Распознавание лиц» и «Персонализатор», чтобы с помощью функции распознавания лица с камеры определить, какой клиент делал заказ, и автоматически предложить пиццу, которая может понравиться этому клиенту.

Машинное обучение было частью приложения LUIS, однако в Azure доступно очень много других ресурсов и инструментов машинного обучения. Возможность обрабатывать крупные наборы данных и вычислять модели данных машинного обучения в высокопроизводительных вычислительных ресурсах Azure снижает входной барьер для тех, кто хочет разрабатывать приложения для работы с достаточно крупными наборами данных. Приложения точнее и эффективнее, и вам не нужно приобретать оборудование или устанавливать специальные инструменты, потому что все необходимые компоненты доступны в DSVM. Не все приложения подходят для работы с искусственным интеллектом и машинным обучением, но по мере того как ваши клиенты начинают рассчитывать на большее, эти сервисы Azure помогут вам выделиться на фоне конкурентов.

Пакетная обработка рабочих нагрузок

Еще 2 области Azure, которые могут представлять интерес с точки зрения больших данных и вычислений для машинного обучения — это пакетный сервис Azure и сервисы для высокопроизводительных вычислений. Пакетный сервис Azure позволяет выполнять крупные и повторяющиеся вычислительные задачи без необходимости управления кластерами планировщиков для работы. Пакетный сервис запускает задачи на виртуальных машинах, используя собственную систему управления и планировщик, — точно так же масштабируемые наборы включают функции автоматического масштабирования и балансировки нагрузки виртуальных машин. Несмотря на то, что пакетный сервис не связан с машинным обучением напрямую, он отлично подойдет, если вам требуется выполнить другие крупные вычислительные задачи.

В Azure также предусмотрены компоненты для высокопроизводительных вычислений, позволяющие работать с крупными виртуальными машинами и осуществлять доступ к виртуальным машинам графических процессоров. Для запуска приложений, требующих большого объема вычислительных ресурсов, можно также использовать определенные инструменты и наборы, такие как DataSynapse и пакет Microsoft HPC.

Машинное обучение, пакетный сервис Azure и высокопроизводительные вычисления — отличные примеры того, как использовать поставщиков облачных вычислений, например Azure, для выполнения крупных вычислительных задач. Вы платите только за используемые вычислительные ресурсы, поэтому вам не нужно покупать и обслуживать дорогостоящее оборудование, которое большую часть времени будет простаивать.

17.4 *Практическое упражнение: добавление каналов для связи с ботом*

В предыдущих примерах вы общались с ботом в текстовом окне на портале Azure. Каналы позволяют расширить варианты взаимодействия с вашим ботом. Можно предоставить боту возможность общаться в Skype, Facebook Messenger и таких приложениях, как Microsoft Teams и Slack. Сервис Azure Bot упрощает пошаговую интеграцию бота с внешними сервисами:

1. На портале Azure выберите бот веб-приложения и нажмите «Каналы».
2. Выберите подходящий канал, например Skype.

   Другие каналы часто требуют создания подключения разработчика (например, Facebook или Slack). В Skype реализована возможность копировать и вставлять HTML-код.
3. Предоставьте необходимую информацию, например код приложения бота. Его можно найти в разделе «Параметры» на странице «Управление ботами».
4. При необходимости используйте онлайн-редактор кода для создания базовой HTML-страницы, например default.htm в каталоге wwwroot, и вставьте необходимый код для своего канала. Открыть веб-приложение можно на портале Azure, а затем выбрать его URL-адрес, чтобы открыть страницу default.htm с кодом канала, например http://azuremol.azurewebsites.net/default.htm.

18 Автоматизация Azure

По возможности не следует выполнять вход на сервер и вносить изменения вручную. Не нужно устанавливать программное обеспечение, нажимая кнопки в графическом интерфейсе пользователя. Кроме того, не нужно вносить обновления в файлы конфигурации в текстовом редакторе. При выполнении таких действий вручную велика вероятность ошибки, что может привести к ошибкам конфигурации и сбоям приложений. Если вам потребуется реплицировать конфигурацию сервера, вы сможете вспомнить все действия, выполненные для того, чтобы запустить в работу существующий сервер? Что если через полгода вам потребуется повторить то же самое?

В главе 16 мы рассматривали автоматическую проверку и применение обновлений для серверов. Все это становится возможным благодаря сервису автоматизации Azure. В этой главе мы рассмотрим создание, запуск и редактирование модулей Runbook, а также использование настройки требуемого состояния PowerShell для автоматической установки приложений и настройки сер веров.

18.1 Что такое сервис автоматизации Azure?

Учетная запись сервиса автоматизации Azure объединяет множество элементов, как показано на рисунке 18.1. Основная функция — это создание и выполнение сценариев по требованию или по определенному расписанию. Можно создать сценарии в PowerShell или Python, а затем предоставить платформе Azure возможность планировать и выполнять модули Runbook. Можно предоставлять учетные данные и объекты подключения, а также автоматически применять требуемые конфигурации серверов и составлять соответствующие отчеты. Сервис управления обновлениями, рассмотренный нами в главе 16, обеспечивает безопасность и актуальность ваших серверов благодаря доступности новейших исправлений и обновлений хоста на протяжении всего жизненного цикла среды приложений.

Рис. 18.1. Сервис автоматизации Azure предоставляет множество связанных функций. Общий набор ресурсов, таких как учетные данные, сертификаты, расписания и объекты подключения, может использоваться для автоматического выполнения скриптов PowerShell или Python на целевых серверах. Можно определить требуемое состояние сервера, а сервис автоматизации Azure установит и настроит сервер соответствующим образом. Обновления хоста и исправления системы безопасности могут применяться автоматически. Все эти функции работают на серверах с Windows и Linux, в Azure, локальной среде и системах других поставщиков облачных технологий.

Чтобы упростить управление несколькими модулями Runbook или настройками требуемых состояний в учетной записи сервиса автоматизации Azure, можно предоставить общий доступ к следующим ресурсам:

- *Расписания* позволяют определить время и периодичность выполнения каждого модуля Runbook или задачи по управлению обновлениями. Если впоследствии нужно изменить периодичность, можно изменить одно из общих расписаний, вместо того чтобы вносить изменения в каждый модуль Runbook или задачу по управлению обновлениями, которые используют эти расписания.
- *Модули* дополняют базовую функциональность, сохраняя дополнительные модули PowerShell. Базовые модули Windows PowerShell и Azure уже доступны, а дополнительные модули, например для управления средой Linux, можно добавить и использовать в модулях Runbook.
- *Учетные данные* для разных учетных записей, которые имеют разрешения на выполнение различных модулей Runbook, сохранены как ресурсы, а не определены в каждом модуле Runbook. Такой подход позволяет при необходимости обновлять и сбрасывать учетные данные, а каждый использующий их модуль Runbook обновляется автоматически. Следовательно, учетные данные не сохраняются в модулях Runbook в виде обычного текста, что повышает безопасность этих модулей.
- *Подключения* определяют свойства аутентификации для субъектов-сервисов Azure AD. Это особый тип учетной записи пользователя, который позволяет модулям Runbook осуществлять доступ к вашим ресурсам Azure. Как правило, для обеспечения дополнительной безопасности такие подключения используют цифровые сертификаты, а не имена пользователя и пароли.
- *Сертификаты* часто интегрируются с ресурсами подключения, чтобы обеспечить безопасный способ проверки удостоверения субъекта-сервиса. Как и в случае с базовыми учетными данными, можно регулярно обновлять эти сертификаты

в центральном расположении, а каждый использующий их модуль Runbook может автоматически получать доступ к новым сертификатам. Можно создать и сохранить собственные сертификаты для использования с модулями Runbook или определениями настроек требуемых состояний.

- *Переменные* представляют собой централизованное расположение для значений среды выполнения: сохраняемых имен, строк местоположения и целых чисел. При выполнении модулей Runbook вставляются эти переменные. Такой подход ограничивает количество жестко запрограммированных ресурсов внутри каждого модуля Runbook.

Работайте интеллектуальнее без лишних усилий

В главе 16 мы говорили о том, как сервисы управления Azure можно использовать для мониторинга серверов в Azure, локальной среде или средах других поставщиков облачных технологий, и составления соответствующих отчетов. Требуемые агенты устанавливаются и настраиваются на удаленных серверах, а затем обеспечивается возможность их подключения к инфраструктуре Azure.

Сервис автоматизации Azure также может работать на разных платформах и в разных инфраструктурах. Так, гибридный рабочий модуль Runbook может выполнять модули Runbook автоматизации на серверах за пределами Azure. Вы по-прежнему используете общие ресурсы автоматизации, которые определяют учетные данные, подключения и сертификаты, но на этот раз эти ресурсы можно использовать для определения компонентов аутентификации для разных платформ. Кроме того, настройки требуемого состояния можно использовать на виртуальных машинах не на платформе Azure (как с Windows, так и с Linux).

В любом случае компонент шлюза устанавливается в удаленной среде, чтобы функционировать в качестве прокси-сервера для отправляемых на целевые объекты команд автоматизации. Благодаря использованию шлюза, функционирующего в качестве прокси-сервера, создается единая точка подключения модуля автоматизации к удаленным средам, а ввиду отсутствия прямого доступа к серверам, которые иначе были бы удаленными, снижаются риски безопасности.

Возможно, для выполнения модулей Runbook и определений настроек требуемых состояний на локальных физических серверах, а не виртуальных машинах Azure, эти модули и определения придется немного изменить. Как и в случае с сервисом архивации Azure, Site Recovery и сервисом управления обновлениями, преимущество сервиса автоматизации Azure заключается в том, что предоставляется единая плоскость управления и набор инструментов для автоматизации процессов в разных инфраструктурах и на разных серверах.

18.1.1 Создание учетной записи сервиса автоматизации Azure

Давайте попробуем создать учетную запись сервиса автоматизации Azure и рассмотрим, какие модули Runbook по умолчанию входят в ее состав. Демонстрационные модули Runbook — отличный фундамент для создания собственных модулей. Кроме того, в графическом редакторе можно перетаскивать составные элементы и создавать сценарии автоматизации.

Попробуйте сейчас

Чтобы создать учетную запись сервиса автоматизации Azure и примеры модулей Runbook, выполните следующие действия:

1. На портале Azure в левом верхнем углу выберите «Создать ресурс».
2. Найдите и выберите «Автоматизация», а затем нажмите «Создать».

   В разделе «Автоматизация и контроль» также создается рабочая область Operations Management Suite (OMS) и настраивается гибридная рабочая роль сервиса автоматизации для управления ресурсами за пределами Azure. Скоро на смену OMS придут сервисы Azure, которые мы рассматривали в предыдущих главах. Прямо сейчас нам нужно создать только ресурс автоматизации.
3. Укажите имя, например azuremol, а затем создайте новую группу ресурсов, например `azuremolchapter18`.
4. Выберите наиболее подходящий (ближайший к вам) регион Azure и оставьте установленным флажок «Создать учетную запись запуска от имени Azure».

   При выборе этого варианта в Azure AD создаются дополнительные учетные записи. Кроме того, создаются сертификаты безопасности для автоматизированной аутентификации учетных записей. Отправлять запросы пользователю и сохранять пароль при этом не требуется. Можно создать и задать дополнительные стандартные учетные данные учетной записи, определенные как ресурс автоматизации, чтобы обеспечить точный контроль над тем, какие учетные записи используются для выполнения определенных модулей Runbook.

В сочетании с управлением доступом на основе ролей (RBAC), рассмотренным в главе 6, можно создать конкретные учетные записи Run as (Запуск от имени) для модулей Runbook и предоставить ограниченный набор разрешений для выполнения конкретных необходимых задач в каждом модуле Runbook или наборе таких модулей. С точки зрения безопасности такой подход позволяет отслеживать и контролировать, как и когда используются эти учетные записи. Не поддавайтесь соблазну создать одну учетную запись запуска от имени, предоставляющую разрешения администраторского уровня, поскольку в этом случае защита от злоупотреблений минимальна.

18.1.2 Ресурсы и модули Runbook сервиса автоматизации Azure

Созданная в разделе 18.1.1. учетная запись сервиса автоматизации Azure включает несколько примеров модулей Runbook. Доступны примеры для PowerShell и Python. Ресурсы подключения и сертификаты также добавляются в учетную запись службы автоматизации для созданных учетных записей запуска от имени. Давайте рассмотрим общие ресурсы подключения подробнее.

Попробуйте сейчас

Чтобы просмотреть настроенные ресурсы и примеры модулей Runbook, выполните следующие действия:

1. На портале Azure щелкните «Группы ресурсов» слева, выберите свою группу, например `azuremolchapter18` и учетную запись сервиса автоматизации Azure, например `azuremol`.
2. Используя раздел «Общие ресурсы» в меню слева, выберите «Подключения».
3. Выберите AzureRunAsConnection, как показано на рисунке 18.2:
4. Выберите «Сертификаты» в главном меню учетной записи сервиса автоматизации в разделе «Общие ресурсы», а затем выберите AzureRunAsCertificate. Как показано на рисунке 18.3, цифровой эскиз соответствует подключению RunAsConnection из предыдущего шага.

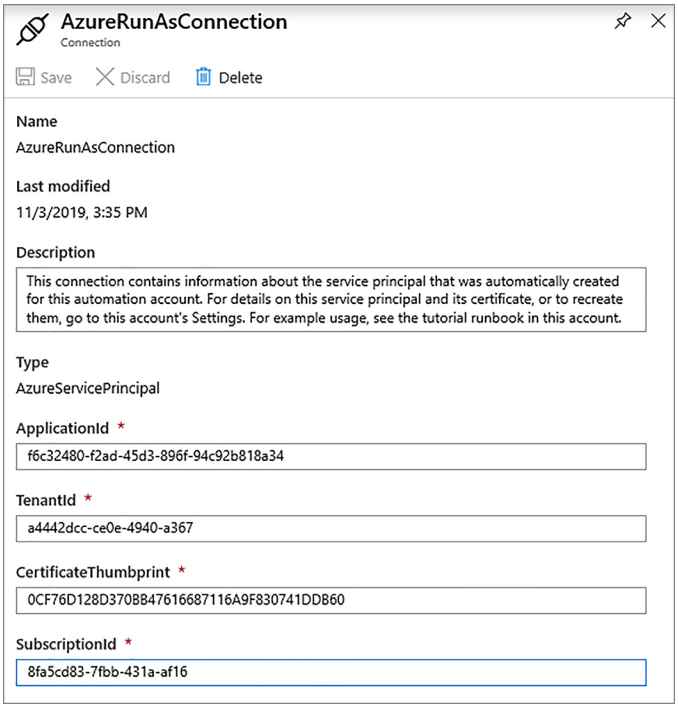


Рис.18.2 Сведения об учетной записи запуска от имени включают в себя ApplicationId и TenantId — специфические свойства для Azure AD, которые помогают определить учетные данные для этой учетной записи. Эскиз CertificateThumbprint соответствует цифровому сертификату, который мы рассмотрим на следующем шаге.

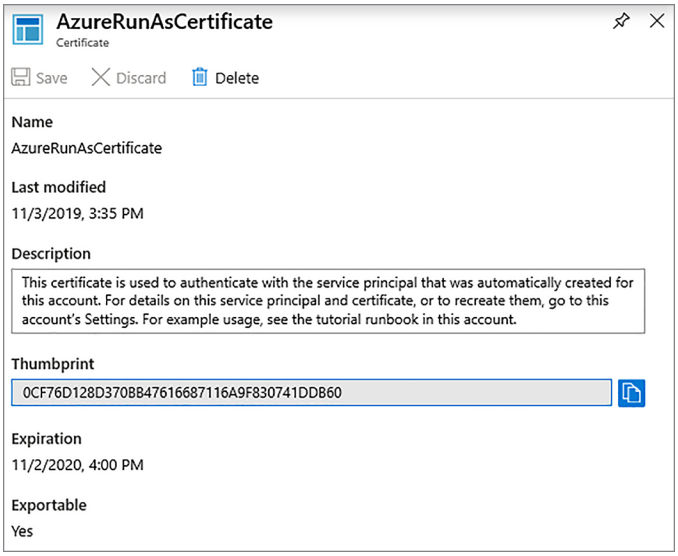


Рис. 18.3. Эскиз RunAsCertificate соответствует показанному в подключении RunAsConnection. В своих модулях Runbook вы определяете, какой ресурс подключения следует использовать. Соответствующий сертификат используется для входа в учетную запись Azure.

5 Теперь, когда вы разбираетесь в ресурсах для подключений и сертификатов, давайте рассмотрим один из примеров модулей Runbook. Выберите «Модули Runbook» в меню слева в учетной записи сервиса автоматизации. Доступно несколько примеров модулей Runbook.

6 Выберите модуль Runbook PowerShell с именем AzureAutomationTutorialScript.

7 Вверху окна с примером модуля Runbook доступны параметры «Пуск», «Просмотр» и «Изменить» для запуска, просмотра и редактирования модуля Runbook. Надеюсь, их функции вам понятны.

Кроме того, доступен параметр «Расписание», позволяющий создать или выбрать общий ресурс, определяющий точное время выполнения модуля Runbook по расписанию, и параметр «Веб-перехватчики», позволяющий создать URL-адрес для выполнения модуля Runbook из другого скрипта или действия. Выберите «Просмотр».

Служба автоматизации Azure и управление версиями с помощью GitHub

Модули Runbook можно интегрировать с системой управления версиями, такой как GitHub. Одним из величайших преимуществ системы управления версиями модулей Runbook является возможность документирования управления изменениями и возврата к предыдущим версиям модулей Runbook при возникновении проблемы.

Всякий раз когда вы сохраняете модуль Runbook службы автоматизации Azure, в системе управления версиями сохраняется новая версия. Для этого не нужно покидать редактор модулей Runbook, поскольку платформа Azure и настроенная система управления версиями работают в обоих направлениях. Если при работе с новым модулем Runbook вы столкнетесь с проблемой, можно запросить в системе управления версиями предыдущую версию, что позволит продолжить выполнять задания без задержки, а позже вы сможете провести диагностику и выяснить, почему в обновленной версии возникла проблема.

Использование системы управления версиями также позволяет получать информацию о том, какие изменения и когда имели место. Если требуется провести аудит ваших модулей Runbook или понять, как они изменялись со временем, системы управления версиями предоставят вам информацию о различиях в версиях.

18.2 *Пример модуля Runbook службы автоматизации Azure*

Рассмотрим, как пример модуля Runbook Powershell — AzureAutomationTutorialScript — подключается к Azure и собирает информацию о ваших ресурсах. Кроме того, при желании вы можете изучить пример модуля Runbook на языке Python — их структуры аналогичны. PowerShell и Python — единственные языки программирования, которые в настоящее время поддерживаются в модулях Runbook службы автоматизации Azure. В следующем фрагменте кода указаны учетные данные для подключения модуля Runbook.

Фрагмент кода 18.1. Настройка учетных данных подключения

```
$connectionName = "AzureRunAsConnection"
try
{
  # Get the connection "AzureRunAsConnection "
  $servicePrincipalConnection=Get-AutomationConnection -Name
➥$connectionName
```

Создает объект для $connectionName

Отправляет запрос на подключение

Создает объект субъектаслужбы

```
    "Logging in to Azure..."
    Add-AzureRmAccount `
        -ServicePrincipal `
        -TenantId $servicePrincipalConnection.TenantId `
        -ApplicationId $servicePrincipalConnection.ApplicationId `
        -CertificateThumbprint
➥$servicePrincipalConnection.CertificateThumbprint
}
```

Выполняет вход в Azure

Код начинается с создания объекта для `$connectionName`. В упражнении «Попробовать» вы видели, что был создан ресурс подключения по умолчанию для AzureRunAsConnection. Создавая собственные модули Runbook, вы, возможно, захотите создать дополнительные учетные записи запуска от имени и ресурсы подключения, чтобы разделить модули Runbook и используемые ими учетные данные. Компоненты подключения и процедура обработки исключений, которые мы рассмотрим далее, должны быть одинаковыми для всех модулей Runbook. При необходимости можно изменить используемый ресурс подключения.

Оператор `try` служит для отправки запроса на подключение. Создается объект субъектаслужбы с именем `$servicePrincipalConnection` на основе `$connectionName`. Затем модуль Runbook выполняет вход в Azure с учетной записью `Add-AzureRmAccount` и использует объект `$servicePrincipalConnection` для получения параметров `TenantId`, `ApplicationId` и `Certificate-Thumbprint`. Мы обсуждали эти параметры в составе ресурса подключения ранее. Затем ресурс сертификата, соответствующий эскизу `$servicePrincipalConnection`, используется для завершения входа в Azure.

В следующем фрагменте кода показано, что если происходит сбой подключения, модуль Runbook перехватывает ошибку и останавливает выполнение.

Фрагмент кода 18.2. Перехват ошибки и остановка выполнения модуля Runbook

```
catch {
     if (!$servicePrincipalConnection)
     {
          $ErrorMessage = "Connection $connectionName not found."
          throw $ErrorMessage
     } else{
          Write-Error -Message $_.Exception
          throw $_.Exception
     }
}
```

Оператор `catch` обрабатывает все ошибки в составе попытки входа в систему. Если подключение субъекта-сервиса найти не удается, выдается ошибка. Как правило, эта ошибка означает, что найти заданный ресурс подключения невозможно. Проверьте имя и правильность написания вашего подключения.

В противном случае объект подключения был бы найден, для входа в систему использовался бы субъект-сервис, но успешно завершить аутентификацию не удалось бы. Сбой мог произойти по причине недействительности сертификата или отключении учетной записи запуска. Эта функциональность показывает, как можно отозвать учетную запись в Azure AD и запретить выполнение модулей Runbook, использующих эти учетные данные.

Теперь модуль Runbook получает список всех ресурсов Azure.

Фрагмент кода 18.3. Получение списка ресурсов Azure

```
$ResourceGroups = Get-AzureRmResourceGroup
foreach ($ResourceGroup in $ResourceGroups)
{
   Write-Output ("Showing resources in resource group "
➥+ $ResourceGroup.ResourceGroupName)
    $Resources = Find-AzureRmResource -ResourceGroupNameContains
➥$ResourceGroup.ResourceGroupName |
➥Select ResourceName, ResourceType
    ForEach ($Resource in $Resources)
    {
        Write-Output ($Resource.ResourceName + " of type "
➥+ $Resource.ResourceType)
    }
    Write-Output ("")
}
```

Код модуля Runbook размещается в заключительной части модуля. Создается объект для `$ResourceGroups`, который получает список всех доступных групп ресурсов Azure. Затем через каждую группу ресурсов проходит цикл `foreach`, который находит список ресурсов и составляет список имен и типов ресурсов.

В этом базовом примере показано, как можно взаимодействовать с Azure после того, как модуль Runbook прошел аутентификацию в подписке. Если вы используете в работе с учетной записью запуска от имени систему управления доступом на основе ролей, возвратятся только группы ресурсов, на просмотр которых у учетной записи есть разрешения. Подобное использование RBAC демонстрирует, почему с точки зрения безопасности целесообразно создавать и использовать учетные записи запуска от имени ограниченной области действия, чтобы ограничить доступ модулей Runbook к ресурсам в вашей среде Azure. Всегда старайтесь предоставить минимальные необходимые привилегии.

Если языки PowerShell и Python вам незнакомы, не волнуйтесь. Они представляют собой прекрасные базовые языки создания скриптов, но могут также использоваться для разработки сложных многофункциональных приложений. Вам как разработчику будет относительно легко освоить и использовать любой из них. Автоматизация задач помогает освободить рабочее время ИТ-специалистов, которое они могут уделить выполнению других важных задач, и отличным подспорьем в этом окажется язык PowerShell или Python. У издательства Manning Publications также есть для вас немало отличных книг!

18.2.1 Запуск примера модуля Runbook и просмотр его выходных данных

Теперь, когда вы узнали, что содержит пример скрипта модуля Runbook и как используются ресурсы подключения и сертификатов, давайте выполним модуль Runbook и посмотрим на результаты.

Попробуйте сейчас

Чтобы увидеть модули Runbook в действии, выполните приведенные ниже действия:

1. Закройте окно, отображающее содержимое модуля Runbook, и вернитесь к обзору AzureAutomationScriptTutorial.
2. Выберите «Пуск» в верхней части окна модуля Runbook.

3 Подтвердите, что вы хотите запустить модуль Runbook, а затем подождите несколько секунд, пока Runbook начнет работать.

4 Выберите «Вывод», как показано на рисунке 18.4, а затем просмотрите окно консоли, когда Runbook выполняет вход в Azure, получает список групп ресурсов, выполняет циклический перебор и выводит список ресурсов в каждом из них.

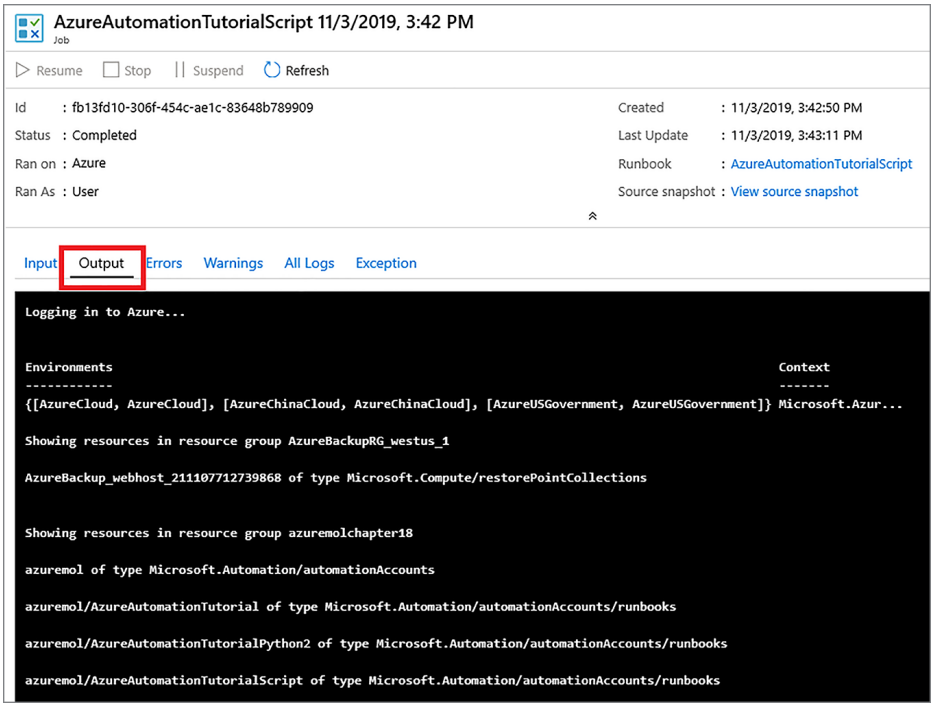


Рис. 18.4. Вы можете просмотреть выходные данные модуля Runbook, а также любые созданные журналы, ошибки и предупреждения. Этот базовый пример будет выполнен за несколько секунд, однако выполнение сложных модулей Runbook занимает больше времени. Вы можете отслеживать выполнении продолжительных модулей Runbook, остановить или временно приостановить их выполнение.

Модули Runbook сервиса автоматизации не должны существовать изолированно. Один модуль Runbook может выполнять другой модуль Runbook. Эта возможность позволяет создавать сложные, многоэтапные процессы автоматизации и минимизировать дублирование кода. Проектируя и создавая модули Runbook, попробуйте разделить их на небольшие дискретные блоки кода. Общие функции с возможностью повторного использования, например вход в систему Azure и составление списка ресурсов или списка виртуальных машин, следует создавать в виде небольших модулей Runbook с возможностью включения в крупные модули. При выпуске новых командлетов PowerShell или изменении параметров вы сможете быстро обновить общий модуль Runbook, который содержит эти командлеты, вместо того чтобы обновлять несколько разных модулей Runbook. На первый взгляд может показаться, что для создания небольших модулей Runbook для многократного использования требуется большей усилий. Но будьте уверены, по мере расширения вашей среды и автоматизации вы скажете мне «спасибо»! Многие задачи в этой книге выполнялись в небольших развертываниях, но начните думать о том, как развертывать приложения и управлять ими в больших масштабах.

18.3 Настройка требуемого состояния (DSC) PowerShell

В главе 12 представлена концепция расширений ВМ. *Расширение* — это небольшой программный компонент, устанавливаемый на виртуальной машине для выполнения поставленной задачи. Расширение диагностики ВМ было установлено на виртуальной машине, чтобы метрики производительности и журналы диагностики возвращались на платформу Azure из виртуальной машины. Это прекрасно, но мы также немного поговорили об автоматической установке программного обеспечения.

Одним из способов установки ПО и настройки сервера является использование настройки требуемого состояния (DSC) PowerShell. DSC позволяет определить, как требуется настроить сервер, то есть его требуемое состояние. Можно определить устанавливаемые пакеты, настраиваемые функции или создаваемые файлы, например. DSC, однако, не ограничивается первоначальной установкой и настройкой. Со временем на серверах часто проводятся обслуживания и операции диагностики, когда конфигурации и пакеты меняют вручную. Затем сервер отклоняется от настроенного вами изначально требуемого состояния. На рисунке 18.5 показано, как сервис автоматизации Azure может выступать в качестве центрального сервера, на котором хранятся определения DSC, позволяя целевым серверам получать их конфигурации и сообщать о соответствии этих серверов требованиям.

Локальный диспетчер конфигураций (LCM) на каждом целевом сервере контролирует процесс подключения к опрашивающему серверу сервиса автоматизации Azure, получение и разбор определения DSC, а также применение конфигураций и отчетность по соответствию. Система LCM может работать без опрашивающего сервера, когда вы локально вызываете процесс чтения и применения определения DSC. В этом режиме,

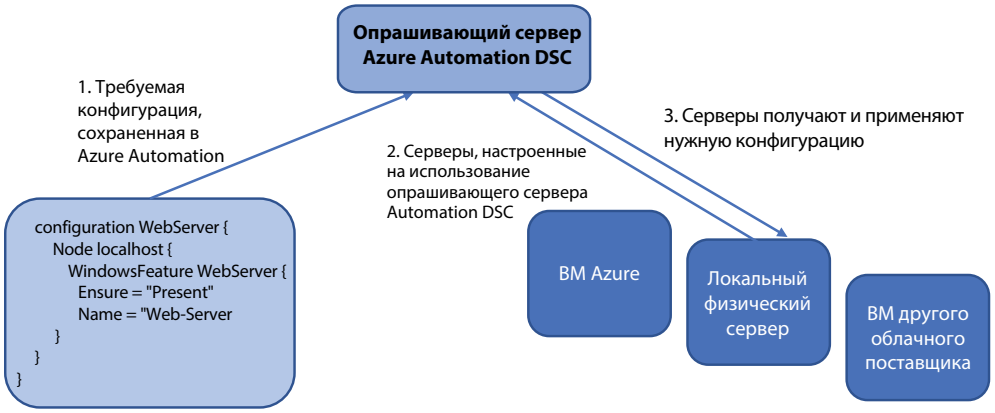


Рис. 18.5. Настройка требуемого состояния для сервера создается и сохраняется в сервисе автоматизации Azure. Учетная запись службы автоматизации действует как опрашивающий сервер, который позволяет подключенным серверам извлекать необходимую конфигурацию из центрального расположения. Можно задать для восстановления сервера разные режимы конфигурации, если их конфигурация отклоняется от требуемого состояния.

где вы вручную перемещаете конфигурацию в систему LCM, вы упускаете много центральных элементов управления и отчетов, которые часто необходимы при управлении большим числом серверов.

Кроме того, существует определенная гибкость в том, как целевые серверы обрабатывают определения DSC, полученные от опрашивающего сервера сервиса автоматизации Azure. DSC можно настроить для работы в одном из трех режимов конфигурации:

- *Только применение* — требуемое состояние отправляется на целевой сервер и применяется на нем, и больше ничего не происходит. Это похоже на поведение расширения пользовательских сценариев Azure в том, что любые конфигурации и установки выполняются при первом развертывании, а процессов, позволяющих предотвратить изменение этих конфигураций вручную на протяжении жизненного цикла сервера, не существует.
- *Применение и мониторинг* — после применения к серверу требуемого состояния DSC продолжает отслеживать изменения, которые заставляют сервер отклоняться от первоначальной конфигурации. Просмотреть серверы, которые более не соответствуют желаемому состоянию, можно в централизованном отчете. Эта конфигурация представляет собой хороший компромисс между необходимостью сохранять на сервере требуемое состояние и вмешательством человека в процесс принятия решений о вариантах исправления проблем.
- *Применение и автоматическая коррекция* — наиболее автоматизированная и автономная конфигурация применяет требуемое состояние, а затем отслеживает отклонения и автоматически исправляет параметры сервера при внесении любых изменений, тем самым обеспечивая его соответствие требованиям. Существует опасность, что нужные изменения вручную будут перезаписаны с целью возврата к настройке требуемого состояния, однако в этом режиме конфигурации гарантируется, что назначенные вами параметры всегда имеют приоритет.

Настройку требуемого состояния PowerShell можно использовать на виртуальных машинах, выполняемых на платформах поставщиков других облачных решений, а также на локальных виртуальных машинах и физических серверах. Благодаря .NET Core настройку требуемого состояния PowerShell можно также использовать на серверах Linux, так что это решение подходит не только для Windows. Поддержка нескольких поставщико в и операционных систем делает PowerShell мощным средством настройки серверов любого масштаба и управления ими.

Можно создать собственный опрашивающий сервер DSC, однако встроенные функции сервиса автоматизации Azure предоставляют ряд дополнительных преимуществ:

- Управление учетными данными осуществляется централизованно, а сертификаты создаются автоматически.
- Обмен данными между опрашивающим сервером DSC и целевым сервером зашифрован.
- Предоставляемые встроенные отчеты обеспечивают соответствие DSC требованиям. Кроме того, реализована интеграция с модулем Log Analytics, что позволяет создавать подробные отчеты и оповещения.

По сути, это экспресс-курс, посвященный настройке требуемого состояния с помощью PowerShell — это мощный функциональный компонент, который активно используется вот уже несколько лет. В сочетании со службой автоматизации Azure DSC позволяет автоматизировать установку и настройку программного обеспечения. Вспомните предыдущие главы, где обсуждались, к примеру, масштабируемые наборы виртуальных машин. Можно применить конфигурацию DSC к масштабируемому набору с помощью сервиса автоматизации Azure, а затем, по мере создания в масштабируемом наборе виртуальных машин каждая из них будет настраиваться с использованием требуемых компонентов и файлов приложения.

18.3.1 Определение и использование PowerShell DSC и опрашивающего сервера сервиса автоматизации Azure

Надеюсь, что экспресс-тур по PowerShell DSC дал вам представление о возможностях этой технологии! Давайте использовать PowerShell DSC для автоматизации установки базового веб-сервера на виртуальной машине.

Попробуйте сейчас

Чтобы увидеть PowerShell DSC в действии, выполните приведенные ниже действия:

1. Создайте виртуальную машину с Windows Server 2019 Datacenter и откройте TCP-порт 80 для HTTP-трафика. Это глава 18, никто больше не будет держать вас за ручку! Вы можете создать виртуальную машину в Cloud Shell или на портале Azure — выбор за вами. Используйте группу ресурсов, созданную в предыдущем упражнении, например `azuremolchapter18`. Пока виртуальная машина развертывается, можете выполнить еще несколько действий.
2. На локальном компьютере создайте файл с именем webserver.ps1, введите следующий код, сохраните и закройте файл, когда все будет готово:

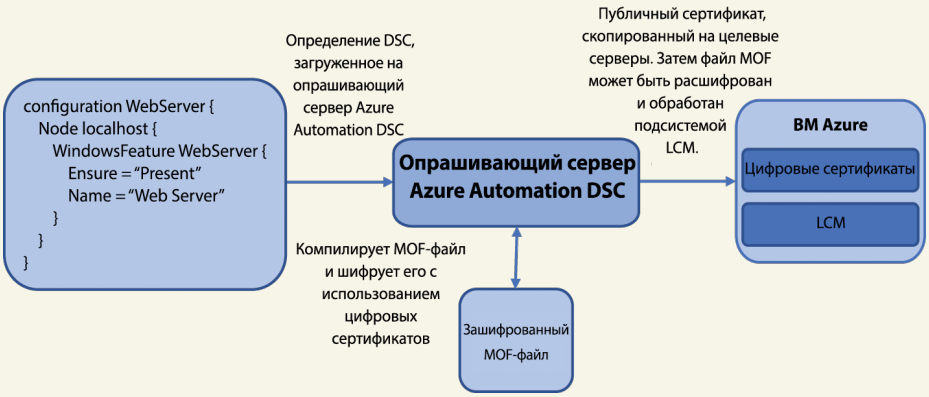
```
configuration WebServer {
    Node localhost {
        WindowsFeature WebServer {
            Ensure = "Present"
            Name = "Web-Server"
        }
    }
}
```

3. На портале Azure выберите свою группу ресурсов, а затем выберите свою учетную запись сервиса автоматизации.
4. Слева выберите «State Configuration (DSC)», откройте вкладку «Конфигурации» и в верхней части окна выберите «Добавить конфигурацию».
5. Найдите и выберите файл webserver.ps1. Имя конфигурации должно совпадать с именем файла, поэтому оставьте имя веб-сервера по умолчанию, а затем нажмите кнопку «ОК».

   Для загрузки и создания конфигурации нужно несколько минут.
6. Когда все будет готово, выберите конфигурацию из списка и нажмите кнопку «Компилировать».

За кулисами DSC

Давайте на минуту остановимся, чтобы поговорить о том, что происходит, когда вы компилируете конфигурацию, как показано на рисунке сбоку. Чтобы распространить определения DSC, файлы PowerShell конвертируются в файлы формата MOF. Этот тип файлов используется не только для PowerShell DSC — он позволяет понятно и централизованно вносить изменения в конфигурацию компонентов Windows. Любое определение DSC, не только в сервисах автоматизации Azure, необходимо компилировать, прежде чем его можно будет применять к целевому серверу. Система LCM принимает и обрабатывает только файлы MOF.



Сервер извлечения DSC сервиса автоматизации Azure автоматически компилирует определение DSC, которое вы предоставляете в MOF-файл. Цифровые сертификаты, используемые службой автоматизации, служат для шифрования файла MOF. Целевые серверы DSC получают необходимые публичные цифровые сертификаты и позволяют системе LCM расшифровать и обработать MOF-файл. Затем требуемое состояние можно применить к серверу.

Поскольку MOF-файл определяет полное состояние ваших серверов, необходимо защитить его содержимое. Если злоумышленник знает все установленные компоненты приложения и расположение различных файлов конфигурации и пользовательского кода, это увеличивает вероятность атаки на ваши серверы. Последние версии PowerShell шифруют весь MOF-файл. Сервис автоматизации Azure автоматически создает необходимые цифровые сертификаты и ключи, когда целевой сервер настроен для DSC, что позволяет беспрепятственно использовать зашифрованные MOF-файлы. Автоматизация также шифрует трафик между опрашивающим сервером DSC и целевыми узлами, а не только MOF-файлом.

Процесс компиляции в сервисе автоматизации Azure преобразует определение DSC, которое вы предоставляете в MOF-файл, и шифрует MOF-файл с помощью цифровых сертификатов и ключей. Процесс компиляции определения DSC занимает несколько секунд, но прекрасно защищает вашу среду — это еще один пример защиты ресурсов Azure по умолчанию!

7 Чтобы применить конфигурацию к ВМ, откройте вкладку «Узлы» в окне «State configuration (DSC)», нажмите кнопку «Добавить» и выберите виртуальную машину, созданную ранее.

8 Выберите «Подключить».

9 В раскрывающемся меню «Имя конфигурации узла» выберите webserver.local host.

10 Выберите режим конфигурации ApplyAndMonitor и нажмите кнопку «ОК».

Чтобы включить использование опрашивающего сервера DSC Azure PowerShell на виртуальной машине и применить первоначальное требуемое состояние, нужно несколько минут.

11 Когда на портале Azure появится сообщение о применении конфигурации, выберите группу ресурсов. Затем выберите ВМ, созданную ранее.

12 Вы открыли TCP-порт 80 для ВМ при ее создании? Если нет, создайте правило группы безопасности сети, чтобы разрешить трафик, а затем откройте публичный IP-адрес ВМ в браузере. В процессе настройки требуемого состояния был установлен веб-сервер IIS, после чего загружается веб-страница по умолчанию, как показано на странице 18.6.

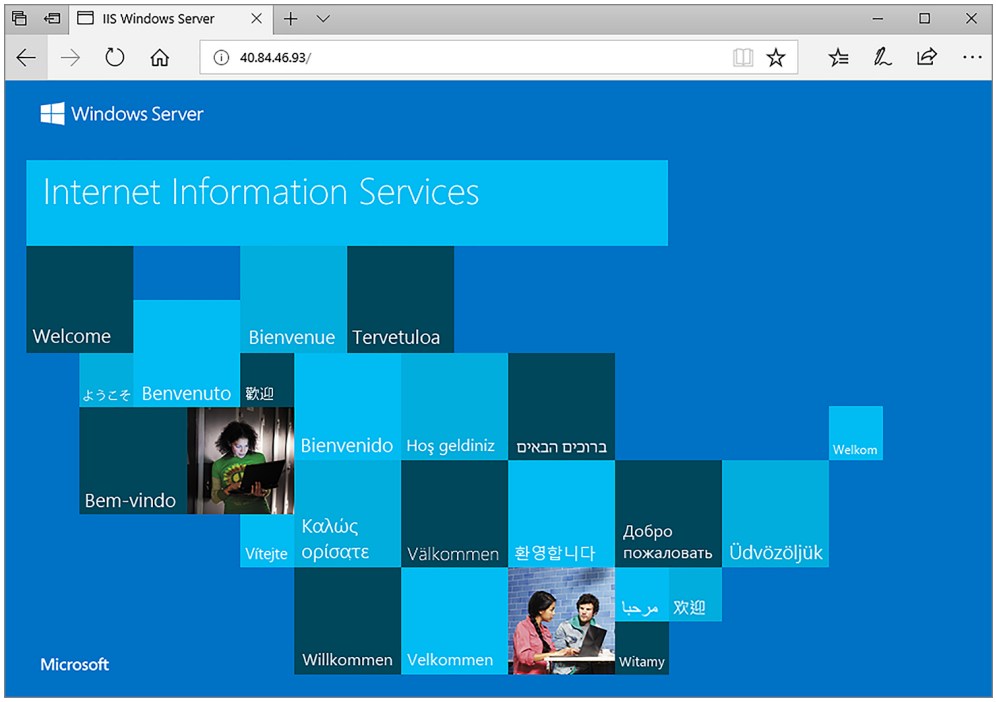


Рис. 18.6. После подключения виртуальной машины к DSC сервиса автоматизации Azure применяется требуемое состояние и устанавливается веб-сервер IIS.

В базовом примере PowerShell DSC устанавливается только веб-сервер. PowerShell DSC можно использовать для настройки веб-сервера IIS или копирования кода приложения на виртуальную машину и запуска сайте. Сложные определения DSC можно использовать, чтобы подготовить виртуальную машину для обслуживания трафика клиентов вашего онлайн-магазина пиццы без вмешательства вручную. Подумайте о том, как спроектировать свои приложения для автоматического масштабирования — виртуальная машина не может ждать, пока кто-нибудь выполнит вход и вручную все установит и настроит!

18.4 *Практическое упражнение: использование DSC с Linux*

Чтобы доказать, что PowerShell DSC работает на серверах Linux, давайте создадим виртуальную машину с Ubuntu, установим необходимые компоненты, а затем установим базовый веб-сервер NGINX с DSC. В производственной среде можно было бы использовать пользовательский образ виртуальной машины с установленными компонентами управления, а затем применить определения PowerShell DSC в обычном режиме:

1. Число дистрибутивов Linux, поддерживаемых PowerShell DSC для Linux без дополнительной настройки, ограничено. Поэтому чтобы упростить это упражнение в конце главы, создайте виртуальную ВМ CentOS 7.7 или поздней версии и откройте порт 80.
2. В учетной записи сервиса автоматизации Azure выберите в меню слева пункт «Модули».
3. Нажмите кнопку «Обзор галереи», а затем найдите, выберите и импортируйте модуль nx для управления ресурсами DSC Linux.
4. На локальном компьютере создайте файл с именем httpd.ps1 и введите следующий код:

```
configuration httpd {
    Import-DSCResource -Module nx
    Node localhost {
        nxPackage httpd {
            Name = "httpd"
            Ensure = "Present"
            PackageManager = "yum"
        }
        nxService httpd {
            Name = "httpd"
            State = "running"
            Enabled = $true
            Controller = "systemd"
        }
    }
}
```

5. Добавьте конфигурацию DSC в учетную запись сервиса автоматизации Azure, отправьте файл httpd.ps1 и скомпилируйте конфигурацию.
6. Добавьте узел DSC в учетную запись сервиса автоматизации Azure, выберите ВМ CentOS, а затем выберите имя конфигурации узла httpd.localhost.

   Опять же, требуется несколько минут, чтобы виртуальная машина загрузила требуемую конфигурацию. Можно просмотреть список подключенных виртуальных машин и их статус соответствия в окне «Узлы DSC». Сообщается, что виртуальная машина соответствует требованиям, если система LCM приняла и применила файл MOF, однако установка требуемых пакетов httpd на ВМ может занять еще пару минут.
7. Выберите ВМ CentOS на портале Azure, получите ее публичный IP-адрес и введите его в браузере, чтобы просмотреть веб-сервер, установленный DSC. Если веб-сайт не загружается, подождите пару минут, пока процесс установки не завершится, а затем обновите страницу.

Если вы действительно хотите погрузиться в совершенно новый мир Microsoft и Linux, вы можете установить PowerShell на вашей виртуальной машине Linux. Выполните описанные на сайте http://mng.bz/VgyP шаги по быстрой установке, чтобы оценить возможности современных скриптов PowerShell с поддержкой нескольких платформ!

19 Контейнеры Azure

Контейнеры Docker и Kubernetes за несколько лет приобрели огромную популярность. Подобно тому, как виртуализация серверов привела к изменению способов, с помощью которых ИТ-отделы управляли своими центрами обработки данных в середине 2000-х годов, современные инструменты и оркестраторы контейнеров теперь меняют способы создания и работы приложений. Развитие контейнеров никак не связано с облачными вычислениями, но в сочетании контейнеры и облачные вычисления обеспечивают отличный способ разработки приложений с использованием облачного подхода. Платформам Docker и Kubernetes посвящены целые книги, но давайте вкратце рассмотрим, как можно быстро запускать контейнеры в Azure. Существует широкий набор сервисов Azure, предназначенных для контейнеров. Он больше соответствует подходу «платформа как сервис». Вы можете сосредоточиться на создании и работе своих приложений, а не на управлении инфраструктурой, оркестрацией и кластерными компонентами контейнеров.

В этой главе мы рассмотрим, что такое контейнеры, как с ними связана платформа Docker и что может сделать для вас Kubernetes. Чтобы узнать, как можно быстро запустить один экземпляр контейнера или несколько экземпляров контейнеров в кластере, мы рассмотрим экземпляры контейнеров Azure (ACI) и сервис Azure Kubernetes (AKC).

19.1 Что такое контейнеры?

В течение нескольких последних лет произошел огромный всплеск интереса к контейнерам, и я удивлюсь, если вы не слышали по крайней мере об одной компании, вызвавшей этот всплеск, — компании Docker. Однако что конкретно представляет собой контейнер, и при чем здесь Docker?

Сначала рассмотрим традиционный хост виртуализации, на котором работают виртуальные машины. Рисунок 19.1 подобен схеме, которая была рассмотрена в главе 1, где каждая виртуальная машина имеет собственное виртуальное оборудование и гостевую ОС.

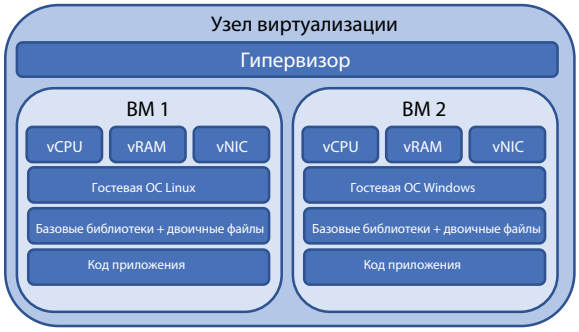


Рис. 19.1. При использовании традиционной инфраструктуры виртуальных машин гипервизор на каждом хосте виртуализации обеспечивает уровень изоляции, предоставляя каждой виртуальной машине собственный набор виртуальных аппаратных устройств — виртуальный процессор, виртуальное ОЗУ и виртуальные сетевые адаптеры. Виртуальная машина устанавливает гостевую ОС, например Ubuntu Linux или Windows Server, которая может использовать это виртуальное оборудование. Наконец, вы устанавливаете свое приложение и все необходимые библиотеки. Благодаря такому уровню изоляции виртуальные машины становятся очень безопасными, но добавляются издержки, связанные с вычислительными ресурсами, ресурсами хранения данных и временем запуска.

В контейнере отсутствует виртуальное оборудование и гостевая ОС. В него включаются только базовые приложения и библиотеки, необходимые для работы приложения (см. рис. 19.2).

В одном гипервизоре может работать много виртуальных машин, каждая из которых имеет собственную виртуальную гостевую ОС, виртуальное оборудование и стек приложений. Гипервизор управляет запросами от виртуального оборудования каждой виртуальной машины, планирует распределение и совместное использование физических аппаратных ресурсов, а также обеспечивает безопасность и изоляцию каждой виртуальной машины. Работа гипервизора представлена на рис. 19.3.

Рис. 19.2. В контейнере содержатся только необходимые для работы приложения базовые библиотеки, двоичные файлы и код приложения. Контейнер — это легкий переносимый элемент, поскольку в нем нет гостевой ОС и уровня виртуального оборудования, что также уменьшает его размер на диске и время запуска.

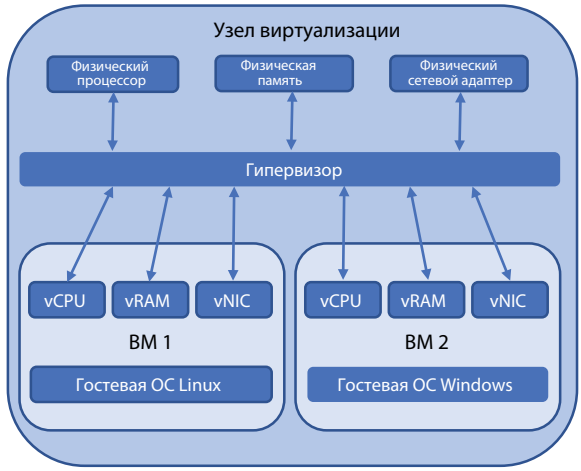


Рис. 19.3. На традиционном хосте ВМ гипервизор обеспечивает планирование запросов от виртуального оборудования каждой виртуальной машины в направлении базового физического оборудования и инфраструктуры. Гипервизор обычно не знает о том, выполнение каких инструкций планирует гостевая ОС на то или иное время работы физического процессора, ему требуется только время этого процессора.

На одном хосте может работать несколько контейнеров. Хост контейнеров получает различные системные вызовы от каждого контейнера и планирует распределение и совместное использование этих запросов в общем базовом ядре, ОС и аппаратных ресурсах. Контейнеры обеспечивают логическую изоляцию процессов приложений. Работа среды выполнения контейнеров представлена на рис. 19.4.

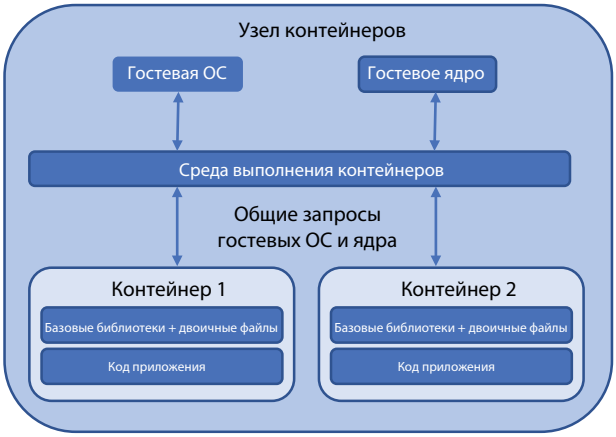


Рис. 19.4. Контейнеры совместно используют общую гостевую ОС и ядро. Среда выполнения контейнеров обрабатывает запросы от контейнеров в направлении общего ядра. Каждый контейнер работает в изолированном пользовательском пространстве, и некоторые дополнительные функции безопасности защищают контейнеры друг от друга.

Контейнеры обычно гораздо легче виртуальных машин. Они запускаются быстрее виртуальных машин, часто за считанные секунды, а не минуты. Размер образа контейнера, как правило, составляет лишь десятки или сотни мегабайт по сравнению с несколькими десятками гигабайт для виртуальных машин. По-прежнему существуют границы зон безопасности и элементы управления безопасностью, но важно помнить, что каждый контейнер использует то же ядро, что и другие контейнеры на том же хосте.

Попробуйте сейчас

Чтобы создать кластер AKS для использования в предстоящих упражнениях, потребуется всего несколько минут, поэтому выполните указанные ниже действия, а затем продолжите чтение данной главы:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Создайте группу ресурсов/ Введите имя, например `azuremolchapter19`, и расположение, например `eastus`. Доступность AKS зависит от региона, поэтому выберите крупный регион, например `eastus` или `westeurope`. (Актуальный список доступности регионов см. на странице https://azure.microsoft.com/regions/services.)

   ```
   az group create --name azuremolchapter19 --location eastus
   ```

3. Чтобы создать кластер Kubernetes, укажите для параметра `--node-count` значение `2`. Используйте масштабируемые наборы и зоны доступности (о которых вы узнали из предыдущих глав):

```
az aks create \
  --resource-group azuremolchapter19 \
  --name azuremol \
  --node-count 2 \
  --vm-set-type VirtualMachineScaleSets \
  --zones 1 2 3 \
  --no-wait
```

Последний параметр `--no-wait` возвращает управление в Cloud Shell до завершения создания кластера. Продолжайте чтение во время развертывания кластера.

Компания Docker присоединилась к разработчикам контейнеров с набором инструментов и стандартных форматов, которые определяют, как создать и запустить контейнер. ПО Docker создается поверх существующих функций уровня ядра Linux и Windows для обеспечения переносимости и согласованности работы контейнеров на разных платформах. Разработчик может создать контейнер Docker на ноутбуке с macOS, проверить свое приложение, а затем запустить этот контейнер без изменений в традиционном кластере серверов под управлением Linux или Windows локально или в Azure. Все необходимые двоичные файлы, библиотеки и файлы конфигурации приложения связываются воедино, становясь составной частью контейнера, поэтому базовая ОС хоста не является важным аспектом или ограничением при разработке.

Следует отметить важность Docker. Термины *контейнер* и *Docker* часто используются взаимозаменяемо, хотя технически это неточно. Docker — это набор инструментов для разработчиков, помогающих создавать и запускать контейнеры согласованным, надежным и переносимым способом. Благодаря простоте использования этих инструментов они быстро стали применяться на практике, а базовая контейнерная технология, которая в той или иной форме существовала на протяжении 10 лет, стала популярной. Разработчики приняли контейнеры и платформу Docker, и с тех пор ИТ-отделам приходится догонять их.

Компания Docker участвует в проекте Open Container Initiative. Формат и спецификации, которые она определила для упаковки и запуска контейнеров, были одними из основополагающих принципов этого проекта. Работа Docker продолжалась и была поддержана другими компаниями. Крупными участниками работ в области создания контейнеров являются компании IBM и Red Hat, которые предоставили результаты некоторых базовых разработок и код, поддерживающий существующие контейнерные платформы. Формат упаковки и сред выполнения контейнеров, определенный Open Container Initiative и при проектировании, имеет важное значение, поскольку благодаря ему каждый поставщик может размещать свои собственные инструменты поверх общих форматов, что позволяет вам перемещать базовый контейнер между платформами и получать при этом одинаковые базовые результаты.

19.2 Подход к разработке и развертыванию приложений с использованием микросервисов

Если контейнеры предлагают концепцию изоляции, аналогичную виртуальным машинам, можно ли в них выполнять те же типы рабочих нагрузок, что и на виртуальных машинах? И да, и нет. То, что вы можете что-то делать, не означает, что вы должны это делать! Контейнеры можно использовать для выполнения любых рабочих нагрузок, с которыми вы знакомы. При этом есть преимущества, касающиеся возможностей переносимости и оркестрации, которые будут рассмотрены в разделе 19.4. Чтобы максимально увеличить преимущества контейнеров и настроить себя на достижение успеха, воспользуйтесь возможностью применить немного другую ментальную модель, когда начнете работать с контейнерами. На рисунке 19.5 сравнивается традиционная модель приложения с одходом на основе использования микросервисов.

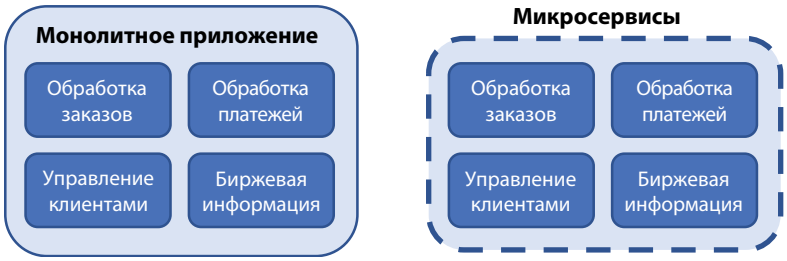


Рис. 19.5. Традиционное монолитное приложение работает как единое целое. Приложение может состоять из различных компонентов, но запускается из одного места установки и исправляется и обновляется как один экземпляр. При использовании микросервисов каждый компонент делится на собственные сервисы приложения и исполняемый блок. Каждый компонент можно обновлять, исправлять и масштабировать независимо от других компонентов.

Стандартная виртуальная машина содержит полностью установленную гостевую ОС, например Ubuntu или Windows Server. Эта базовая ОС включает в себя сотни компонентов, библиотек и инструментов. После ее установки можно установить дополнительные библиотеки и приложения, например для веб-сервера NGINX или Microsoft SQL Server. Наконец, развертывается код приложения. Такая виртуальная машина обычно выполняет большую часть приложения, если не все. Это одно большое приложение и один запускаемый экземпляр. Для повышения производительности в виртуальную машину можно добавить дополнительную память или процессор (вертикальное масштабирование, рассмотренное в предыдущих главах) либо увеличить количество экземпляров, выполняющих приложение (горизонтальное масштабирование, как в случае масштабируемых наборов). Создание нескольких экземпляров приложения возможно только в том случае, если приложение поддерживает кластеры. При создании нескольких экземпляров приложения часто используется некоторая форма общего хранилища для обеспечения согласованного состояния экземпляров. Приложение, развернутое таким традиционным способом, называется *монолитным*.

Другой подход к проектированию, разработке и выполнению приложений состоит в разделении приложений на мелкие компоненты. Это подход к разработке и развертыванию приложений с использованием *микросервисов*. Каждый микросервис отвечает за небольшую часть широкой среды приложения. Микросервисы можно расширять, масштабировать и обновлять независимо от остальной среды приложения.

Хотя сначала, пока разработчики и ИТ-специалисты учатся применять другой способ создания и развертывания приложений, эта модель может создавать проблемы, контейнеры отлично подходят для концепции использования микросервисов. Разработчики могут развернуть инкрементные обновления меньшего размера быстрее, чем при использовании монолитного подхода к разработке приложений. Микросервисы и контейнеры также отлично подходят для рабочих процессов непрерывной интеграции и непрерывной поставки, в ходе которых легче создавать, тестировать, поэтапно готовить и развернуть обновления. Ваши клиенты получают новые функции или исправления ошибок быстрее, и в результате ваша компания растет.

Микросервисы с Azure Service Fabric

В этой главе основное внимание уделяется контейнерам Docker и оркестрации с помощью Kubernetes, но есть еще один сервис Azure, который похож своим способом перемещения разработки приложений в направлении модели с микросервисами.

Платформа Azure Service Fabric существует уже несколько лет и исторически представляла ориентированный на Windows подход к созданию приложений, при котором каждый компонент делится на собственные микросервисы. Service Fabric отслеживает, где каждый компонент микросервиса работает в кластере, позволяет сервисам обнаруживать друг друга и взаимодействовать между собой, а также управляет резервированием и масштабированием.

Платформа Service Fabric используется многими крупными сервисами Azure, такими как Cosmos DB. Это говорит от том, что она очень функциональна и эффективна. Сама платформа Service Fabric работает поверх масштабируемых наборов виртуальных машин. Вы ведь знаете кое-что о масштабируемых наборах, верно?

Платформа Service Fabric стала функциональнее и теперь может работать как с Windows, так и с Linux в качестве гостевой ОС, поэтому приложения можно создавать на любом удобном для вас языке программирования. В Azure также можно выбрать способ управления и оркестрации контейнерных приложений. Как платформа Service Fabric, так и AKS обладают большими преимуществами и их можно применять в разных сценариях использования.

Если вы в настоящее время разрабатываете или хотели бы разрабатывать микросервисы вне контейнеров, платформа Service Fabric отлично подойдет в качестве отправной точки. Приложения, разработанные на базе модели акторов, также отлично подходят, так как платформа Service Fabric создавалась с учетом этой модели программирования. Она обеспечивает единый подход для работы как с традиционными приложениями для микросервисов, так и с приложениями на основе контейнеров. Если затем вы решите применять контейнеры для других рабочих нагрузок, вы можете использовать те же инструменты и интерфейс управления Service Fabric для управления всеми средами приложений.

Для использования ориентированного на контейнеры подхода к приложениям с самого начала лучше выбрать AKS, поскольку развитие и внедрение Kubernetes обеспечивают первоклассную работу контейнеров. В AKS можно работать с контейнерами Linux и Windows.

19.3 Экземпляры контейнеров Azure

Теперь, когда вы немного больше узнали о том, что такое контейнеры и как их можно использовать, давайте углубимся в эту тему и создадим основной экземпляр магазина пиццы. Это пример из предыдущих глав, где вы создали базовую виртуальную машину, которая обеспечивала работу вашего веб-сайта, или развернули приложение в веб-приложениях. В обоих случаях вам требовалось создать виртуальную машину или веб-приложение, подключиться к этой виртуальной машине или веб-приложению, а затем развернуть там основную веб-страницу. Могут ли возможности контейнеров существенно упростить вашу жизнь? Конечно!

Превосходный сервис под названием «Экземпляры контейнеров Azure» (ACI) позволяет создавать и запускать контейнеры за считанные секунды. Не требуется предварительно создавать и настраивать никакие сетевые ресурсы, и каждый экземпляр контейнера тарифицируется посекундно. Если вы никогда не использовали контейнеры и не хотите ничего устанавливать локально на своем компьютере, ACI — отличный способ попробовать эту технологию.

Чтобы увидеть, как можно быстро запустить магазин пиццы, давайте создадим экземпляр контейнера. Для запуска экземпляра контейнера требуется всего одна команда, но на рис. 19.6 показано, как объединить множество компонентов, чтобы все происходило за кадром. Мы рассмотрим компоненты файла Dockerfile и центра Docker после запуска экземпляра контейнера.

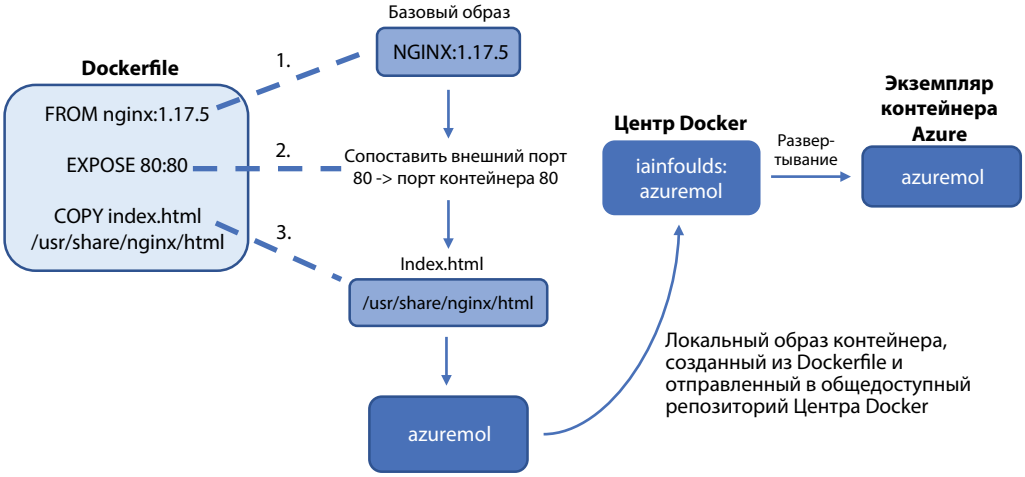


Рис. 19.6. Для создания полного образа контейнера azuremol использовался файл Dockerfile. Этот образ был помещен в сетевой публичный реестр, называемый центром Docker. Затем с помощью этого предварительно созданного публичного образа из центра Docker, который предоставляет готовый к запуску образ приложения, можно создать экземпляр контейнера.

Попробуйте сейчас

Чтобы создать экземпляр контейнера Azure, в котором работает основной веб-сайт, выполните следующие действия:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Создайте экземпляр контейнера, укажите, что вам нужен публичный IP-адрес, и откройте порт 80:

```
az container create \
  --resource-group azuremolchapter19 \
  --name azuremol \
  --image iainfoulds/azuremol \
  --ip-address public \
  --ports 80
```

   В этом упражнении используется пример образа, который я создал для вас. Мы рассмотрим его подробнее, когда контейнер будет запущен.

3. Чтобы увидеть, что было создано, изучите выходные данные команды создания контей нера.

   В разделе «События» можно увидеть, как извлекается (скачивается) образ из Docker Hub, а также создается и затем запускается контейнер.

   Кроме того, назначаются некоторые резервы для процессора и памяти, которые при необходимости можно скорректировать. Отображается публичный IP-адрес, а также некоторые сведения о контейнере, в частности состояние подготовки, тип ОС и политика перезапуска.

4 Чтобы открыть основной веб-сайт, который работает в контейнере, можно запросить только назначенный публичный IP-адрес:

```
az container show \
  --resource-group azuremolchapter19 \
  --name azuremol \
  --query ipAddress.ip \
  --output tsv
```

5 Откройте в веб-браузере веб-страницу по публичному IP-адресу экземпляра контейнера. Должна отобразиться основная веб-страница магазина пиццы, показанная на рис. 19.7.

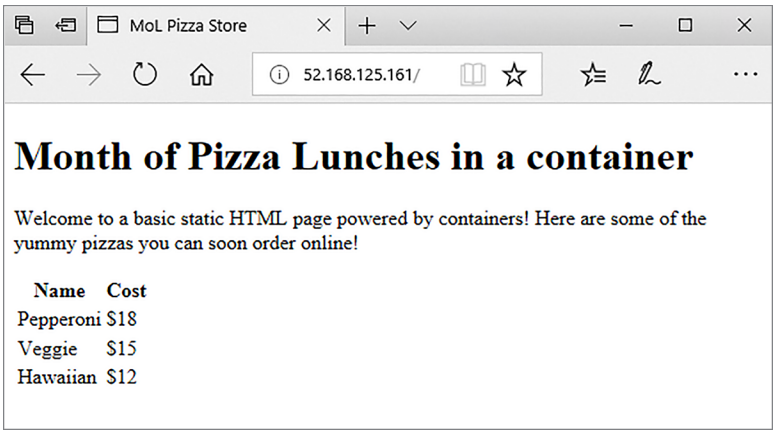


Рис. 19.7. При создании экземпляра контейнера веб-сайт магазина пиццы запускается без дополнительной настройки. Вся конфигурация и содержимое включаются в образ контейнера. В этом упражнении демонстрируется переносимость и эффективность контейнеров: после того как образ контейнера подготовлен, ваше приложение запускается сразу после развертывания нового экземпляра контейнера.

Рассмотрим образ контейнера. Я не хочу вдаваться в подробности Docker и того, как создавать образы контейнеров, но важно понимать, откуда взялся этот образ и как он запускает веб-сайт без дополнительной настройки.

Образ создается из определения конфигурации, называемого *Dockerfile*. В Dockerfile вы определяете базовую платформу, любую конфигурацию, которую хотите применить, и любые команды для запуска или файлы для копирования. Файлы Dockerfile могут быть и часто сложнее, чем в следующем примере, который использовался для создания контейнера azuremol:

```
FROM nginx:1.17.5

EXPOSE 80:80

COPY index.html /usr/share/nginx/html
```

Когда этот Dockerfile использовался для создания образа контейнера Docker, в качестве исходного образа использовался образ NGINX, и пример веб-страницы был скопирован в него. Затем этот контейнер был отправлен в центр Docker — сетевой публичный репозиторий, который Docker предоставляет длясовместного использования и развертывания контейнеров. Чтобы развернуть экземпляр контейнера, в качестве образа контейнера для использования вы указали iainfoulds/azuremol. Платформа Azure обратилась в центр Docker и обнаружила репозиторий с именем iainfoulds, а в нем — образ с именем azuremol.

Рассмотрим строки Dockerfile.

- `FROM nginx:1.17.5`. В предыдущих главах вы создали базовую виртуальную машину, подключились к ней по протоколу SSH, а затем вручную установили веб-сервер NGINX. В примере Dockerfile все это сделано в одной строке. Эта строка указывает, что следует создать контейнер на основе существующего образа контейнера, который предварительно установлен с помощью NGINX. `1.17.5` — это версия публичного образа контейнера NGINX, которую мы будем использовать. На момент написания книги это была последняя версия образа. Рекомендуется добавить указанный номер версии. Если этого не сделать, всегда используется последняя версия. В теории это звучит неплохо, но приложения на основе микросервисов могут масштабироваться до множества активных контейнеров, поэтому чтобы ваша среда соответствовала вашему приложению, следует контролировать точный номер версии каждого используемого компонента.
- `EXPOSE 80:80`. Чтобы предоставить доступ к виртуальной машине в предыдущих главах, вы создали правило NSG, разрешающее использовать порт 80. В Dockerfile эта строка указывает контейнеру открыть порт 80 и сопоставить его внутреннему порту 80. При создании экземпляра контейнера с помощью `az container create` вы также указали, что платформа Azure должна разрешать прохождение трафика на порт 80 с помощью строки `--ports 80`. Все это относится к организации виртуальных сетей, о которой вам необходимо думать!
- `COPY index.html /usr/share/nginx/html`. Заключительная часть — размещение вашего приложения в контейнере. В предыдущих главах вы использовали Git для получения примера веб-страницы магазина пиццы, а затем направляли этот пример в веб-приложение. При использовании Dockerfile файл index.html просто копируется командой `COPY` в локальный каталог /usr/share/nginx/html контейнера. Вот и все!

В своих собственных сценариях вы можете определить файл Dockerfile, который использует другой базовый образ, например Node.js или Python. Затем следует установить дополнительные необходимые библиотеки или пакеты поддержки, извлечь код приложения из системы управления версиями, например GitHub, и развернуть приложение. Этот файл Dockerfile будет использоваться для создания образов контейнеров, которые затем будут сохраняться в частном реестре контейнеров, а не в публичном репозитории центра Docker, как в данном примере.

Реестр контейнеров Azure

Возможности Docker Hub могут показаться вам превосходными. Есть ли в Azure что-то подобное? Да! Так как вам нужно создать Dockerfile и образ контейнера, к сожалению, это упражнение займет больше 2 минут, а в этой главе и без этого есть о чем рассказать. Вы можете легко интегрировать реестр контейнеров Azure (ACR) и AKS, так что эти 2 сервиса прекрасно работают друг с другом. Однако вы можете создавать свои собственные образы из Dockerfile в Cloud Shell, и я советую вам изучить, как это делается, если у вас есть время. Что же касается реестра контейнеров Azure (ACR), то это сервис, которы я выбрал для хранения своих образов контейнеров по нескольким причинам:

- Это частный реестр для образов контейнеров, поэтому вам не нужно беспокоиться о возможном нежелательном доступе к файлам и конфигурации вашего приложения. Вы можете применять механизмы RBAC, которые были рассмотрены в главе 6. RBAC позволяет ограничивать доступ к образам и проверять, кто имеет доступ к ним.
- При хранении образов контейнеров в реестре в Azure образы находятся в тех же центрах обработки данных, что и инфраструктура, используемая для работы экземпляров или кластеров контейнеров (что будет рассмотрено в раделе 19.4.1.). Хотя образы контейнеров должны быть относительно небольшими (часто размером лишь в десятки мегабайт), их размер может увеличиваться, если продолжать скачивать эти образы из удаленного реестра.

ACR также предоставляет встроенные функции репликации и резервирования, которые можно использовать для размещения контейнеров почти там, где они развертываются и запускаются для доступа пользователей. Такой способ размещения в регионе аналогичен способу использования глобальной репликации Cosmos DB в главе 10 для учета миллисекунд и предоставления клиентам максимально быстрого доступа к приложениям.

Если все это вам интересно, посмотрите, как быстро ACR начинает работать с вашим собственным частным репозиторием уже через несколько минут: http://mng.bz/04rj.

19.4 Сервис Azure Kubernetes

Запуск одного экземпляра контейнера — это прекрасно, но один экземпляр не обеспечивает большой избыточности и возможности масштабирования. Помните предыдущие главы этой книги, которые были посвящены запуску нескольких экземпляров приложения, распределению нагрузки и автоматическому масштабированию экземпляров? Разве не замечательно сделать то же самое с контейнерами? Вот где требуется оркестратор контейнеров.

Как следует из названия, *оркестратор контейнеров* управляет экземплярами контейнеров, отслеживает их работоспособность и может масштабироваться по мере необходимости. Оркестраторы могут выполнять гораздо больше задач (и часто делают это), но на высоком уровне. Их основная задача — управление всеми динамическими компонентами, участвующими в работе высокодоступного масштабируемого контейнерного приложения. Существует несколько оркестраторов контейнеров, например Docker Swarm и операционная система Distributed Cloud Operating System (DC/OS), но один из них опередил остальные и стал лучшим — это Kubernetes.

Проект Kubernetes начался как проект с открытым исходным кодом под руководством компании Google, которая также спонсировала его, и был основан на ее внутренних средствах оркестрации контейнеров. Этот проект, широко признанный сообществом разработчиков ПО с открытым исходным кодом, является одним из крупнейших и наиболее быстро развивающихся проектов с открытым исходным кодом на портале GitHub. Многие крупные технологические компании, включая Red Hat, IBM и Microsoft, вносят свой вклад в базовый проект Kubernetes.

Давайте воспользуемся в этом разделе примером веб-приложения из предыдущего упражнения с ACI для запуска масштабируемого развертывания с избыточностью в Kubernetes. В итоге мы получим несколько компонентов, показанных на рис. 19.8.

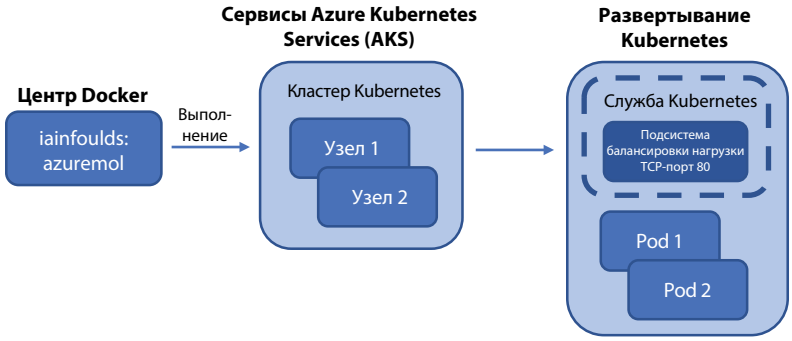


Рис. 19.8. Пример контейнера из центра Docker запускается в кластере Kubernetes с 2 узлами, созданном в AKS. В развертывании Kubernetes есть 2 логических элемента pod (по одному для каждого узла кластера), в каждом из которых работает экземпляр контейнера. Затем вы предоставляете публичную подсистему балансировки нагрузки, позволяющую просматривать ваше веб-приложение в Интернете.

19.4.1 Создание кластера с сервисами Azure Kubernetes

В главе 9 было рассмотрено, как масштабируемые наборы виртуальных машин снижают сложность развертывания и настройки базовой инфраструктуры. Вы только указываете, сколько экземпляров виртуальных машин требуется иметь в масштабируемом наборе, а остальная часть сети, хранилище и конфигурация развертываются автоматически. AKS работает таким же образом, предлагая отказоустойчивый масштабируемый кластер Kubernetes под управлением платформы Azure. Масштабируемые наборы могут использоваться для базовых ВМ, работающих в кластере AKS. Эти ВМ могут быть распределены между зонами доступности. Используются подсистемы балансировки нагрузки Azure, также избыточные в пределах зоны. По сути, AKS объединяет несколько компонентов инфраструктуры и рекомендации, о которых вы узнали из этой книги!

Попробуйте сейчас

Чтобы просмотреть информацию о кластере AKS, выполните следующие действия:

1. Откройте портал Azure и в верхнем меню выберите значок Cloud Shell.
2. Ранее в этой главе вы создали кластер Kubernetes. Процесс создания занял несколько минут, но, надеюсь, сейчас кластер готов! Выясните состояние кластера, введя следующую команду:

```
az aks show \
  --resource-group azuremolchapter19 \
  --name azuremol
```

   Свойство `provisioningState` должно сообщать о состоянии `Succeeded`.
3. Если кластер готов, получите файл учетных данных, который позволяет использовать инструменты командной строки Kubernetes для аутентификации и управления ресурсами:

```
az aks get-credentials \
  --resource-group azuremolchapter19 \
  --name azuremol
```

Вот и все, что нужно, чтобы настроить и запустить Kubernetes в Azure! Вы можете спросить: «Разве нельзя просто создать кластер с виртуальными машинами или масштабируемыми наборами и вручную установить те же компоненты Docker и Kubernetes?» Конечно, можно. Наряду с концепцией использования веб-приложений используются концепции IaaS и PaaS виртуальных машин. Концепция использования веб-приложений предоставляет множество преимуществ: вы задаете только параметры конфигурации высокого уровня, а затем загружаете код приложения. Управляемый кластер Kubernetes, предлагаемый AKS, снижает уровень сложности и упрощает управление — вы концентрируетесь только на своих приложениях и обслуживании клиентов.

Точно так же, как можно выбрать виртуальные машины вместо веб-приложений, можно развернуть собственный кластер Kubernetes, а не использовать AKS. Это нормально, так как в обоих подходах в конечном итоге используются одни и те же компоненты сервисов Azure. Вопросы, связанные с виртуальными машинами, масштабируемыми наборами, подсистемами балансировки нагрузки и группами безопасности сети, были рассмотрены в предыдущих главах, и все они остаются актуальными при рассмотрении кластеров AKS, но не связаны с ними непосредственно. С точки зрения планирования и устранения неполадок вам необходимо иметь соответствующие знания, чтобы понимать, что происходит внутри, и обеспечить надлежащую работу управляемой платформы Kubernetes. В процессе принятия решений при создании нового приложения на основе контейнеров в Azure вы будете действовать в зависимости от того, насколько вам комфортно работать, и времени, которое хотите потратить на управление инфраструктурой.

19.4.2 Запуск базового веб-сайта в Kubernetes

В разделе 19.4.1. был создан кластер Kubernetes, но не были запущены никакие приложения. Давайте изменим эту ситуацию! Теперь необходимо выполнить развертывание Kubernetes, которое было представлено ранее на рис. 19.8; см. рис. 19.9.

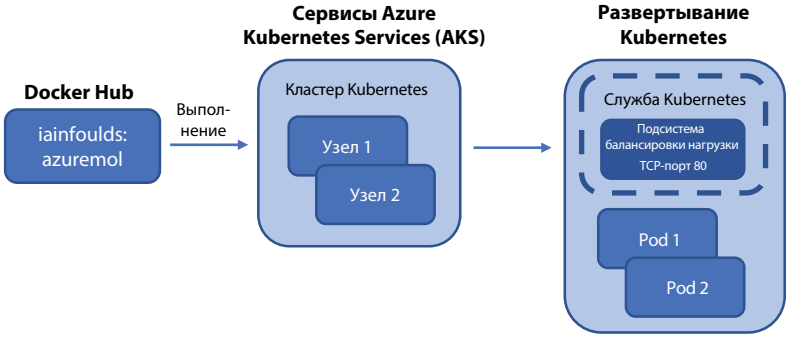


Рис. 19.9. Сейчас в кластере Kubernetes, созданном в AKS, можно выполнить развертывание Kubernetes и запустить приложение. Контейнер работает в обоих узлах с одним логическим элементом pod в каждом узле; требуется создать сервис Kubernetes, который предоставляет публичную подсистему балансировки нагрузки для направления трафика в приложение.

Попробуйте сейчас

Чтобы развернуть приложение в кластере Kubernetes, выполните следующие действия:

1. Взаимодействие с кластером Kubernetes осуществляется с помощью программы командной строки `kubectl`. Используйте тот же образ контейнера iainfoulds/azuremol из Docker Hub, который был запущен в качестве экземпляра контейнера:

```
kubectl run azuremol \
  --image=docker.io/iainfoulds/azuremol:latest \
  --port=80
```

   Для скачивания образа контейнера из центра Docker и запуска приложения в Kubernetes может потребоваться около минуты. Приложение запускается в элементе *pod* — логической конструкции в Kubernetes, где находятся все контейнеры.

2. Элементы pod могут содержать дополнительные вспомогательные компоненты, но сейчас определите состояние контейнера, отобразив состояние элемента pod:

```
kubectl get pods --watch
```

Даже если состояние элемента pod — `Running`, доступ к приложению будет невозможен. Созданный ранее экземпляр контейнера может направлять трафик через публичный IP-адрес непосредственно в этот экземпляр, но что, по вашему мнению, необходимо кластеру Kubernetes, чтобы направлять трафик в контейнеры? Если вы догадались, что требуется подсистема балансировки нагрузки, поздравляю! Сейчас у вас есть только один элемент pod — один экземпляр контейнера. Вы выполните горизонтальное масштабирование элементов pod в практическом упражнении в конце этой главы, но чтобы полученный результат работал, вам потребуется направлять трафик в несколько экземпляров. Давайте укажем, чтобы для решения этой задачи сервис Kubernetes использовал подсистему балансировки нагрузки.

Вот где интеграция между Kubernetes и Azure становится замечательной. Когда вы сообщаете Kubernetes, что хотите создать для своих контейнеров подсистему балансировки нагрузки, Kubernetes возвращается в платформу Azure и создает подсистему балансировки нагрузки Azure. Эта подсистема похожа на подсистему, о которой говорилось в главе 8. Существуют внешние и внутренние пулы IP-адресов и правила балансировки нагрузки, а также можно настроить зонды работоспособности. По мере увеличения или уменьшения масштаба развертывания Kubernetes подсистема балансировки нагрузки при необходимости автоматически обновляется.

Попробуйте сейчас

Чтобы предоставить доступ к вашему приложению из Интернета, выполните следующие действия:

1. Сообщите Kubernetes, что вы хотите использовать подсистему балансировки нагрузки, и добавьте правило для распределения трафика на порту 80:

```
kubectl expose deployment/azuremol \
  --type="LoadBalancer" \
  --port 80
```

2. Как и раньше, определите состояние развертывания сервиса, используя параметр watch:

```
kubectl get service azuremol --watch
```

Присвоение публичного IP-адреса означает, что подсистема балансировки нагрузки Azure завершила развертывание, и кластер и узлы Kubernetes подключены.

3 Введите публичный IP-адрес сервиса в браузере, чтобы увидеть запущенное веб-приложение.

Развертывание приложений в Kubernetes часто намного сложнее, чем показано в этом базовом примере. Обычно определяется манифест службы, аналогичный шаблону Resource Manager, который содержит все характеристики приложения, в частности количество запускаемых экземпляров приложения, присоединяемое хранилище, способы балансировки нагрузки, используемые сетевые порты и т. д. В реальном мире вы не делаете это вручную. Система CI/CD, такая как Azure DevOps или Jenkins, автоматизирует развертывание приложений и сервисов напрямую внутри кластера AKS. Удобство использования AKS состоит в том, что не нужно думать об установке и настройке Kubernetes. Как и при использовании других PaaS-сервисов, таких как веб-приложения и Cosmos DB, вы предоставляете свои приложения и позволяете платформе Azure управлять базовой инфраструктурой и резервированием.

Содержание в чистоте и порядке

Не забывайте очищать и удалять группы ресурсов, чтобы не тратить много средств на счете Azure. Когда вы начинаете изучать контейнеры, очень важно обратить внимание на то, какие ресурсы Azure оставлены включенными. Одно веб-приложение стоит недорого, но кластер AKS из пяти узлов и несколько экземпляров контейнеров с георепликированными образами реестра контейнеров Azure наверняка окажутся дорогими!

Экземпляры ACI тарифицируются посекундно, и, если они работают несколько дней или недель, стоимость быстро растет. В кластере AKS для каждого узла запускается виртуальная машина, поэтому в случае увеличения масштаба и запуска в кластере большого количества виртуальных машин вам придется платить за одну виртуальную машину для каждого узла.

Плата за количество контейнеров, запускаемых каждым из узлов AKS, не взимается, но, как и в случае с любой виртуальной машиной, узел AKS становится дорогим, если его оставить работающим. Удобство использования Kubernetes состоит в том, что конфигурации сервисов (определение элементов pod, подсистем балансировки нагрузки, автоматического масштабирования и т. д.) можно экспортировать для развертывания в другом месте. При создании и тестировании приложений не требуется оставлять кластер AKS работающим. Его можно развернуть при необходимости, а затем развернуть сервис из предыдущей конфигурации.

Кластеры AKS можно масштабировать с уменьшением и увеличением, как вы увидите в практических упражнениях в конце главы. Вы также можете настроить автоматическое масштабирование в зависимости от нагрузки. Это аналогично автомасштабированию, которое мы рассматривали в главе 9 для масштабируемых наборов и веб-приложений. Вы начинаете понимать, что на платформе Azure все объединяется?

В этой главе вы вкратце ознакомились с контейнерами и Kubernetes, поэтому не беспокойтесь, если сейчас чувствуете себя немного ошеломленными! Есть несколько замечательных книг издательства Manning, таких как *Learn Docker in a Month of Lunches* Элтона Стоунмана (Elton Stoneman) (https://livebook.manning.com/book/learn-docker-in-a-month-of-lunches) и *Kubernetes in Action*

(продолжение)

Марко Луксы (https://livebook.manning.com/book/kubernetes-in-action), с подробными сведениями о Docker, разработке приложений для микросервисов и платформе Kubernetes. Если эта глава заинтересовала вас, и вы хотите получить дополнительную информацию, обратитесь к этим книгам.

В примерах в этой главе использовались виртуальные машины Linux для узлов кластера AKS, а затем запускались контейнеры Linux для NGINX. Контейнеры становятся сложнее с точки зрения того, что контейнеры Linux можно запускать только на узлах Linux, например. Как вы узнали в начале главы, контейнеры совместно используют гостевую ОС и ядро. Поэтому вы не можете запускать контейнеры Windows на узле Linux. Как правило, вы также не можете запускать контейнеры Linux на узле Windows. Это связано с техническими деталями, но в общем случае контейнер и базовая ОС узла должны совпадать.

Что замечательно в AKS, так это то, что вы можете запускать как узлы Linux, так и Windows, а значит и контейнеры Linux и Windows! Вам нужно обратить внимание на то, как эти разные контейнеры запланированы на узлах с разными ОС, но этот подход значительно расширяет число приложений и сервисов, которые можно запускать в AKS.

19.5 Практическое упражнение: масштабирование развертывания Kubernetes

В базовом примере в этой главе был создан кластер Kubernetes из двух узлов и один элемент pod, запускающий ваш веб-сайт. В этом практическом упражнении вы узнаете, как можно масштабировать кластер и количество экземпляров контейнеров. Это базовый пример, но чем больше узлов, тем больше экземпляров контейнеров можно запустить. Что еще полезнее, если вам нужно выполнять в кластере больше приложений.

1. Узнать, сколько узлов находится в кластере Kubernetes, можно с помощью команды `kubectl get nodes`. Увеличьте масштаб кластера до 3 узлов:

```
az aks scale \
  --resource-group azuremolchapter19 \
  --name azuremol \
  --node-count 3
```

   Для увеличения масштаба и добавления дополнительного узла требуется 1–2 минуты.

2. Введите команду `kubectl` еще раз, чтобы увидеть состояние узлов. При вертикальном масштабировании узла Kubernetes не создает дополнительные экземпляры контейнеров для приложений автоматически. Поэтому добавление вычислительных ресурсов нового узла не дает какого-то преимущества сразу же.
3. Просмотрите текущее развертывание, введя команду `kubectl get deployment azuremol`. Ранее был создан только один экземпляр. В этом примере приложения не используется новый узел, добавленный в кластер на шаге 1. Масштабируйте его до 5 экземпляров или *реплик*:

```
kubectl scale deployment azuremol --replicas 5
```

4 Снова воспользуйтесь командой `kubectl`, чтобы проверить развертывание. Просмотрите элементы pod, выполняющие экземпляры контейнеров, введя команду `kubectl get pods`. За считанные секунды все эти дополнительные реплики были запущены и подключены к подсистеме балансировки нагрузки.

5 Выполните команду `kubectl get pods -o wide`, чтобы узнать, на каких узлах работают модули pod. Посмотрите на последнюю цифру в имени узла, которая указывает, какой узел в масштабируемом наборе используется. Модули pod должны быть распределены по всем узлам кластера. Так как другие приложения будут масштабировать число контейнеров аналогичным образом, можно начать использовать вычислительные ресурсы на всех узлах кластера по максимуму.

20 Azure и Интернет вещей

Для меня одним из самых интересных направлений в области технологий в последние несколько лет является Интернет вещей (IoT). Я пока не уверен в необходимости подключения к Интернету посудомоечной машины или холодильника, а также в существовании реальных проблем с конфиденциальностью при использовании телевизора или аудиоустройства, которые постоянно подключены к Интернету и все время слушают вас, чтобы выдать ту или иную команду. Однако существует много практических областей применения устройств Интернета вещей. Вы можете иметь производственное оборудование, которое сообщает о своем состоянии, создает оповещения о необходимости технического обслуживания и позволяет операторам понимать, насколько эффективно оно работает на разных заводах по всему миру. Автотранспортная компания может передавать телеметрические данные со своих транспортных средств о перевозимых грузах и среднем времени вождения, а также разумно изменять маршруты движения водителей по мере необходимости. Судоходные компании могут отслеживать все контейнеры и помогать своим клиентам лучше управлять своей цепочкой поставок, зная, где находятся их ресурсы.

В Azure устройства Интернета вещей можно интегрировать с целым рядом сервисов. Веб-приложения Azure могут предоставлять внешний интерфейс для визуализации данных, хранилище можно использовать для регистрации данных, передаваемых от устройств, а бессерверные функции, например сервис Azure Logic Apps (рассматриваемый в следующей, последней, главе), могут обрабатывать полученные данные.

В этой главе мы рассмотрим, что такое Интернет вещей и как использовать центр Интернета вещей Azure для централизованного управления данными от устройств и сбора этих данных. Затем вы увидите, как с помощью веб-приложения Azure можно просматривать данные от устройства Интернета вещей в режиме реального времени.

20.1 Что такое Интернет вещей?

Интерес к Интернету вещей (IoT) значительно вырос за последние несколько лет, но это расплывчатый термин, который можно применять для многих сценариев. На базовом уровне Интернет вещей — это концепция подключения множества взаимосвязанных устройств (обычно небольших недорогих электронных) к центральным системам и приложениям. Подключенные устройства обычно передают информацию, которую

они собирают от присоединенных датчиков или входов. Эта информация затем обрабатывается центральной системой — возможно, с помощью искусственного интеллекта или машинного обучения, как описано в главе 17, — и приводит к выполнению соответствующих действий. На рис. 20.1 представлена общая структура Интернета вещей.

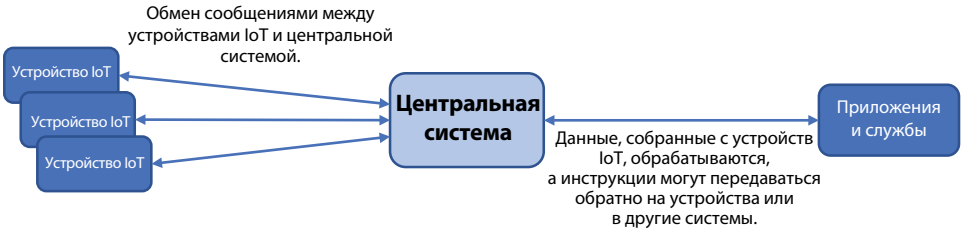


Рис. 20.1. Между множеством подключенных устройств Интернета вещей и центральной системой передаются сообщения. Приложения и сервисы обрабатывают полученные данные и отправляют устройствам инструкции для выполнения дополнительных действий в соответствии с собранными данными.

Ниже приведены некоторые реальные примеры Интернета вещей:

- *Крытая автостоянка*. Небольшой датчик над каждым машиноместом определяет, припаркован ли автомобиль. Если машиноместо пустое, индикатор над ним может гореть зеленым светом, а если оно занято — красным. Водители, въезжающие на автостоянку, видят в режиме реального времени на каждом этаже информационное табло, на котором указывается, сколько есть свободных парковочных мест. Красный и зеленый индикаторы над каждым машиноместом помогают водителям быстро определять, где есть свободные места, когда они проезжают по проходу.
- *Завод*. Оборудование в заводском цехе может сообщать о выпускаемой продукции, расходных материалах и потребностях в техническом обслуживании. Получив соответствующую информацию, центральная система может назначить технического специалиста для профилактического ремонта оборудования или пополнения запасов расходных материалов, что сокращает время простоя производственной линии. В сочетании с искусственным интеллектом и машинным обучением графики технического обслуживания могут прогнозироваться, в результате чего соответствующее количество материалов или сырья может доставляться на производственную линию непосредственно к моменту возникновения на ней потребности в этих материалах или сырье.
- *Транспорт*. В автобусах и поездах общественного транспорта могут быть установлены датчики GPS, которые сообщают о местоположении и скорости движения. Также может собираться информация о билетах для получения сведений о количестве перевозимых пассажиров. В результате на информационных табло на железнодорожных и автовокзалах могут в режиме реального времени отображаться сведения о времени прибытия всех транспортных средств. В сочетании с искусственным интеллектом и машинным обучением это позволяет предоставлять ожидающим пассажирам предложения по альтернативным маршрутам в зависимости от условий движения, задержек и количества пассажиров.

Интернет вещей часто работает вместе с другими приложениями и службами. На заводах и общественном транспорте могут использоваться искусственный интеллект и машинное обучение для улучшения информирования о производственных решениях, а также предоставления предложений пассажирам. Веб-приложения могут использовать

сведения, получаемые от устройств Интернета вещей, для предоставления доступа с мобильных устройств или создания оповещений и уведомлений. Данные, получаемые от устройств Интернета вещей, могут регистрироваться в системе базы данных, например Azure Cosmos DB, а затем обрабатываться приложениями бизнес-аналитики и использоваться для создания отчетов.

В перспективе возможны, например, следующие варианты использования Интернета вещей. Ваш холодильник может определять степень заполненности продуктами питания и создавать список необходимых покупок или даже заказывать продукты в местном продовольственном магазине. Ваш автомобиль может сообщать данные в автосалон, чтобы при сдаче автомобиля на обслуживание все необходимые детали и расходные материалы были наготове. Ваша кофемашина может включаться при срабатывании будильника утром, чтобы быть готовой к завтраку.

Большой проблемой при использовании Интернета вещей является безопасность устройств. Так как многие устройства находятся за пределами вашей основной сетевой инфраструктуры и часто подключены к публичному Интернету, подготовка, обслуживание и обновление этих устройств — сложная задача. Многие устройства Интернета вещей маломощные, содержат простую электронику и могут не иметь возможностей хранения и обработки для автоматического обновления своих приложений и средств безопасности, как это делают традиционные настольные компьютеры и ноутбуки. Недостаточно просто развернуть несколько устройств Интернета вещей, особенно устройств потребительского уровня, не имея плана их надлежащей защиты, обновления и обслуживания.

Эти проблемы безопасности не должны препятствовать созданию приложений и сервисов, использующих устройства Интернета вещей. Интернет вещей ставит новые задачи перед традиционным обслуживанием устройств, но есть решения для централизованной подготовки и обслуживания устройств, а также защиты взаимодействия с ними.

Теперь уже я уверен, что вы догадались, что в Azure есть такое решение для Интернета вещей! Платформа предлагает целый набор сервисов IoT. Рассмотрим, как вы можете начать работу с IoT в Azure.

Ускорение развертывания Интернета вещей Azure

Эта глава посвящена центру Интернета вещей Azure — сервису, с помощью которого можно подготавливать и подключать устройства Интернета вещей для создания собственных решений. Можно задавать способы подключения устройств Интернета вещей, указывать, какие пользователи и приложения могут получать доступ к данным этих устройств, обеспечивать безопасные подключения, а также определять способы создания и развертывания инфраструктуры приложений для объединения всех ресурсов.

Акселераторы решений Интернета вещей Azure — это готовые ключевые сценарии, например для удаленного мониторинга устройств или подключенного завода. Акселераторы развертывают широко распространенные сервисы Azure — центр Интернета вещей, веб-приложения, Cosmos DB и службу хранилища — и запускают пример приложения, объединяющего все эти сервисы.

Вам необходимо настроить приложение для своей среды, используемых устройств Интернета вещей, а также данных, которые требуется собирать и отслеживать. Акселераторы решений Интернета вещей предоставляют отличную основу для начала работы. В то время как центр Интернета вещей определяет способ подключения устройств Интернета вещей к Azure, а затем предоставляет вам возможность развернуть необходимые дополнительные сервисы, акселераторы решений Интернета вещей развертывают готовые решения, которые используют наиболее распространенные из применяемых вами сервисов Azure.

Если после чтения этой главы вы захотите получить подробные сведения об Интернете вещей, акселераторы решений Интернета вещей Azure помогут вам узнать

о возможностях, которые предлагает Azure. Как уже отмечалось в этой книге, Azure — это больше, чем просто одна или две независимые сервисы. Существует множество сервисов, которые можно развернуть вместе, чтобы максимально повысить удобство взаимодействия клиентов с приложениями.

20.2 Централизованное управление устройствами с помощью центра Интернета вещей Azure

Центр Интернета вещей Azure позволяет централизованно управлять данными от устройств Интернета вещей, обновлять и передавать эти данные. С помощью этой службы можно настраивать маршруты приложений для данных, получаемых от устройств, подготавливать сертификаты для обеспечения безопасной связи и управлять ими, а также выполнять мониторинг работоспособности с использованием средств диагностики и метрик Azure. Устройства Интернета вещей можно подключать к другим службам и приложениям Azure, чтобы они могли отправлять и получать данные в рамках более широкого решения. Доступом, как и всеми ресурсами в Azure, можно управлять с помощью механизмов RBAC, а для устранения неполадок и мониторинга или создания оповещений можно осуществлять централизованный сбор данных. На рис. 20.2 показано подключение устройств Интернета вещей к различным сервисам и приложениям Azure через центр Интернета вещей, выступающий в качестве центрального сервиса.

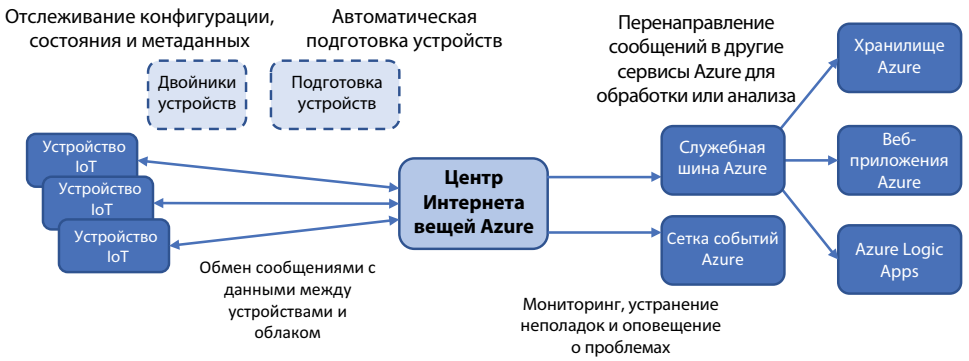


Рис. 20.2. С помощью центра Интернета вещей можно централизованно подготавливать множество устройств Интернета вещей и управлять ими в нужном масштабе. Между устройствами и Azure существует двусторонняя связь для чтения и записи данных. Данные, получаемые от устройств, можно обрабатывать и направлять в другие сервисы Azure, например в веб-приложения и хранилище. Для отслеживания и устранения неполадок информацию можно направлять в сетку событий Azure (которая будет рассмотрена в главе 21), а затем связывать с другими решениями для мониторинга.

Доступом к центру Интернета вещей можно управлять с помощью политик общего доступа. Эти политики похожи на учетные записи и разрешения пользователей. Существуют политики по умолчанию, позволяющие устройствам и службам подключаться к центру Интернета вещей или считывать информацию из реестра устройств и записывать ее в этот реестр, который отслеживает подключенные устройства Интернета вещей и ключи безопасности. Каждой политике может быть назначено одно или несколько из следующих разрешений:

- Чтение реестра
- Запись в реестр

- Подключение служб
- Подключение устройств

Для подключения к центру Интернета вещей приложения и сервисы используют ключи общего доступа. Как и для хранилища, рассмотренного в главе 4, ключи общего доступа позволяют определять строки подключения для идентификации хоста, политики доступа и ключа доступа. Строка подключения содержит ключ доступа, тип политики доступа и имя хоста центра Интернета вещей. Вот пример строки подключения к центру Интернета вещей:

```
HostName=azuremol.azure-devices.net;SharedAccessKeyName=registryRead;
➥SharedAccessKey=6be2mXBVN9B+UkoPUMuwVDtR+7NZVBq+C7A1xCmQGAb=
```

Существуют первичный и вторичный ключи, которые можно менять и обновлять в целях безопасности аналогично регулярному обновлению паролей. Отслеживать и хранить эти ключи для приложений, чтобы получать их при необходимости, можно с помощью различных решений, например Azure Key Vault (это решение было рассмотрено в главе 15). Такой подход к управлению ключами позволяет часто менять ключи доступа без необходимости обновления всего кода приложения.

Цифровые сертификаты могут храниться в центре Интернета вещей и автоматически подготавливаться для устройств Интернета вещей. Помните, что устройства Интернета вещей часто находятся за пределами вашей базовой инфраструктуры и могут подключаться непосредственно через Интернет без использования механизмов безопасного сетевого подключения, например VPN. Обеспечьте шифрование всех данных, передаваемых между устройствами и центром Интернета вещей, с помощью SSL/TLS. Azure Key Vault может создавать и хранить SSL-сертификаты, которые затем добавляются в центр Интернета вещей. Также можно использовать существующий центр сертификации для запроса и выдачи сертификатов. Важно обеспечить шифрование всех данных, передаваемых между устройствами Интернета вещей и Azure. Иначе вы, скорее всего, увидите сообщение об ошибке.

Маршруты центра Интернета вещей позволяют отправлять данные от устройств Интернета вещей в другие сервисы Azure. Вы можете определять критерии (например, содержимое сообщений с определенным ключевым словом или значением), а затем направлять сообщения для хранения в хранилище Azure или обработки веб-приложением. В одном из следующих упражнений моделируется базовый датчик температуры, подключенный к устройству Интернета вещей. В центре Интернета вещей можно определить маршрут для просмотра входящих данных и, если зарегистрированная температура превысила 30 °C, направить данные в приложение логики для отправки оповещения по электронной почте. Удивительный мир бессерверных вычислений и приложений логики будет рассмотрен в главе 21/

Жизнь на периферии

Эта глава посвящена сервису «Центр Интернета вещей Azure». Другой сервис — Azure IoT Edge — позволяет запускать некоторые сервисы, например «Функции Azure» и Stream Analytics, в локальной среде. Вместо передачи данных всеми устройствами Интернета вещей для централизованной обработки в Azure данные можно обрабатывать локально.

Azure IoT Edge запускает приложения и сервисы в контейнерах (см. главу 19). Использование контейнеров делает IoT Edge портативным и согласованным сервисом на разных устройствах и в различных средах. Можно развернуть готовые сервисы Azure или написать собственные приложения и распространить их на периферийные объекты.

Основное преимущество использования IoT Edge заключается в снижении объема обрабатываемых данных и данных, передаваемых по сети. При наличии возможности локальной обработки данных в IoT Edge большие фрагменты данных можно объединять в пакеты и передавать в Azure. Центральные приложения могут в этом случае собирать

информацию от других периферийных объектов для обработки сервисами ИИ и машинного обучения.

Azure IoT Edge также можно успешно использовать на удаленных объектах, часто встречающихся в нефтегазовой и транспортной отраслях. На таких объектах подключение к Интернету может быть недостаточно надежным для передачи всех данных устройств Интернета вещей в Azure для централизованной обработки. IoT Edge позволяет таким удаленным объектам продолжать работать с некоторой степенью автономии даже при отсутствии подключения к Интернету.

При планировании инфраструктуры приложений, в которую входят устройства Интернета вещей, изучите способы устранения сбоев в сети и проблем, связанных с плохими подключениями к Интернету. Если ваша среда зависит от Интернета, предусмотрите резервные подключения к Интернету и оборудование для маршрутизации данных. Можете также рассмотреть вариант использования IoT Edge для локальной обработки данных, если обрабатывать данные централизованно в Azure невозможно.

Попробуйте сейчас

Чтобы начать работу с Интернетом вещей и создать центр Интернета вещей, выполните следующие действия:

1 Откройте портал Azure, запустите Cloud Shell и создайте группу ресурсов, например `azuremolchapter20`:

```
az group create --name azuremolchapter20 --location eastus
```

2 Вы проделали большую работу с Azure CLI в этой книге, потому что команды Cloud Shell и CLI позволяют быстро создавать ресурсы и управлять ими. Как упоминалось в предыдущих главах, Azure CLI может также использовать дополнительные модули, называемые *расширениями*. Эти расширения добавляют дополнительные функциональные возможности и часто обновляются независимо от обычного цикла выпуска основного интерфейса командной строки Azure. Интернет вещей Azure быстро расширяется и добавляет новые возможности, поэтому основные команды для взаимодействия с центром Интернета вещей предоставляются из расширения Azure CLI.

Чтобы получить полную функциональность, необходимую для последующих упражнений, установите расширение Azure CLI IoT:

```
az extension add --name azure-cli-iot-ext
```

3 Создайте центр Интернета вещей и введите имя, например `azuremol`. В этих упражнениях можно использовать центр Интернета вещей уровня «Бесплатный», f1:

```
az iot hub create \
  --resource-group azuremolchapter20 \
  --name azuremol \
  --sku f1 \
  --partition-count 2
```

ПРИМЕЧАНИЕ. ДЛЯ КАЖДОЙ ПОДПИСКИ МОЖНО СОЗДАТЬ ТОЛЬКО ОДИН ЦЕНТР УРОВНЯ «БЕСПЛАТНЫЙ», НО ТАКИЕ ЦЕНТРЫ ОТЛИЧНО ПОДХОДЯТ ДЛЯ ТЕСТИРОВАНИЯ СВЯЗИ МЕЖДУ УСТРОЙСТВАМИ И ИНТЕГРАЦИИ С ДРУГИМИ СЕРВИСАМИ AZURE. ЦЕНТР БЕСПЛАТНОГО УРОВНЯ ПОЛЬЗОВАНИЯ СЕЙЧАС ПОДДЕРЖИВАЕТ ДО 8000 СООБЩЕНИЙ В ДЕНЬ И МАКСИМУМ 500 ПОДКЛЮЧЕННЫХ УСТРОЙСТВ. ЭТО ЗНАЧЕНИЕ МОЖЕТ

ПОКАЗАТЬСЯ БОЛЬШИМ, НО В ЗАВИСИМОСТИ ОТ ТОГО, ЧТО ВЫ ДЕЛАЕТЕ, ОДНО УСТРОЙСТВО, КОТОРОЕ ОТПРАВЛЯЕТ СООБЩЕНИЕ В ЦЕНТР ИНТЕРНЕТА ВЕЩЕЙ ПРИМЕРНО КАЖДЫЕ 12 СЕКУНД, ПРЕВЫСИТ ЭТО ОГРАНИЧЕНИЕ!

Центр Интернета вещей сейчас довольно пустой. Без подключенных устройств Интернета вещей (одного или нескольких) с ним мало что можно делать. Для Интернета вещей обычно используется устройство Raspberry Pi. Это недорогой миникомпьютер, который может подключаться к сетям Wi-Fi и использовать готовые стандартные датчики температуры, влажности и давления. Его также можно использовать для управления небольшими двигателями, индикаторами и таймерами. Вам не нужно спешить и покупать Raspberry Pi для работы с центром Интернета вещей — это устройство можно смоделировать в веб-браузере!

20.3 Создание имитации устройства Raspberry Pi

Устройства Интернета вещей замечательны, но есть препятствие для начала работы, потому что требуются реальные устройства, так ведь? Нет! Есть несколько способов моделирования устройств Интернета вещей с помощью программного обеспечения. Такой программный подход позволяет сосредоточиться на быстром создании приложения, а затем перейти к использованию реального оборудования. Вам все же требуется обратить внимание на то, как ваш код работает на реальном оборудовании Интернета вещей, особенно на устройствах с низким энергопотреблением, поскольку у них может не быть доступа ко всем необходимым библиотекам или даже ресурсам памяти, как у смоделированного приложения.

Microsoft предоставляет бесплатный симулятор Raspberry Pi на портале GitHub (https://azure-samples.github.io/raspberry-pi-web-simulator). Raspberry Pi отлично подходит для тестирования, но будьте осторожны при использовании дешевого готового оборудования, такого как Raspberry Pi, в производственных средах. Планируйте способы обновления и администрирования таких устройств. Выделенные устройства Интернета вещей, такие как Azure Sphere (https://azure.microsoft.com/services/azure-sphere), предоставляют дополнительные параметры безопасности и управления. Raspberry Pi служит хорошей альтернативой для этой книги, тестирования и обучения. В этом симуляторе доступен распространенный датчик BME280, собирающий показания температуры и влажности, моделируется программным обеспечением, моделирующим также светодиод, который показывает, когда устройство передает данные в центр Интернета вещей. Возможности настройки ограничены, но позволяют увидеть, как базовое приложение Node.js может работать на устройстве Raspberry Pi, получать данные от датчика и отправлять их в Azure.

ПРИМЕЧАНИЕ. ЕСЛИ ТАКИЕ ВЕЩИ, КАК RASPBERRY PI, ЭЛЕКТРОНИКА И ДАТЧИКИ ТЕМПЕРАТУРЫ, А ТАКЖЕ ПРИЛОЖЕНИЕ NODE.JS ПУГАЮТ ВАС, НЕ БЕСПОКОЙТЕСЬ. КАК И В ГЛАВАХ, ПОСВЯЩЕННЫХ ИСКУССТВЕННОМУ ИНТЕЛЛЕКТУ И МАШИННОМУ ОБУЧЕНИЮ, КОНТЕЙНЕРАМ И KUBERNETES, МЫ НЕ БУДЕМ УГЛУБЛЯТЬСЯ В УСТРОЙСТВА ИНТЕРНЕТА ВЕЩЕЙ И ПРОГРАММИРОВАНИЕ. ЕСЛИ К КОНЦУ ЧТЕНИЯ ЭТОЙ ГЛАВЫ У ВАС ПОЯВИТСЯ ЖЕЛАНИЕ ПОДКЛЮЧИТЬ ПАЯЛЬНИК И ПОЭКСПЕРИМЕНТИРОВАТЬ С ЭЛЕКТРОНИКОЙ, ОТЛИЧНО, МОЖЕТЕ ЗАНЯТЬСЯ ЭТИМ!

Прежде чем использовать симулятор Raspberry Pi, необходимо создать определение устройства в центре Интернета вещей Azure. При этом создается уникальный идентификатор устройства, чтобы центр Интернета вещей понимал, с каким устройством он взаимодействует, и как следует обрабатывать данные. В сложных сценариях можно предоставить для устройства дополнительные параметры и выдать цифровые сертификаты. В этом упражнении просто создается удостоверение устройства.

Попробуйте сейчас

Чтобы создать имитацию устройства Интернета вещей Raspberry Pi, выполните следующие действия:

1 В Azure Cloud Shell создайте удостоверение устройства в центре Интернета вещей, например `azuremol`, и введите имя устройства, например `raspberrypi`:

```
az iot hub device-identity create \
--hub-name azuremol \
--device-id raspberrypi
```

2 Помните политики общего доступа из раздела 20.2? Каждое устройство Интернета вещей также имеет собственный ключ доступа и строку подключения, которые используются для его идентификации при взаимодействии с центром Интернета вещей. Эта важная функция Azure IoT защищает устройства и сводит к минимуму риск заражения в случае компрометации одного устройства.

Чтобы использовать устройство с симулятором Raspberry Pi, потребуется информация для строки подключения устройства. Этот уникальный идентификатор содержит имя хоста центра Интернета вещей, идентификатор устройства и ключ доступа:

```
az iot hub device-identity show-connection-string \
  --hub-name azuremol \
  --device-id raspberrypi \
  --output tsv
```

3 Скопируйте содержимое строки подключения. Оно понадобится на шаге 4. Выходные данные будут выглядеть следующим образом:

```
HostName=azuremol.azure-devices.net;DeviceId=raspberrypi;
➥SharedAccessKey=oXVvK40qYYI3M4u6ZLxoyR/PUKV7A7RF/JR9WcsRYSI=
```

4 Теперь самое интересное! Откройте симулятор Raspberry Pi в браузере: https://azure-samples.github.io/raspberry-pi-web-simulator. Посмотрите на раздел кода справа в симуляторе. Рядом со строкой 15 должна находиться переменная `connectionString`, которая уже запрашивает у вас строку подключения для устройства центра Интернета вещей *[Your IoT hub device connection string]*. Скопируйте и вставьте строку подключения из шага 3 (см. рис. 20.3).

5 Нажмите кнопку «Выполнить» под окном кода, чтобы запустить симулятор.

Каждые 2 секунды в окне консоли будет отображаться сообщение с данными, отправляемыми в центр Интернета вещей. Красный светодиод на монтажной схеме также будет при этом мигать, чтобы имитировать возможность контроля выходов, подключенных к Raspberry Pi. Выходное сообщение в окне консоли аналогично следующему:

```
Sending message: {"messageId":1,"deviceId":"Raspberry Pi Web
➥Client","temperature":24.207095037347923,
➥"humidity":69.12946775681091}
```

Откуда поступали показания температуры и влажности? Это устройство является имитацией Raspberry Pi, реального датчика BME280 нет, поэтому эти значения генерируются программно приложением. Если посмотреть на остальную часть кода в

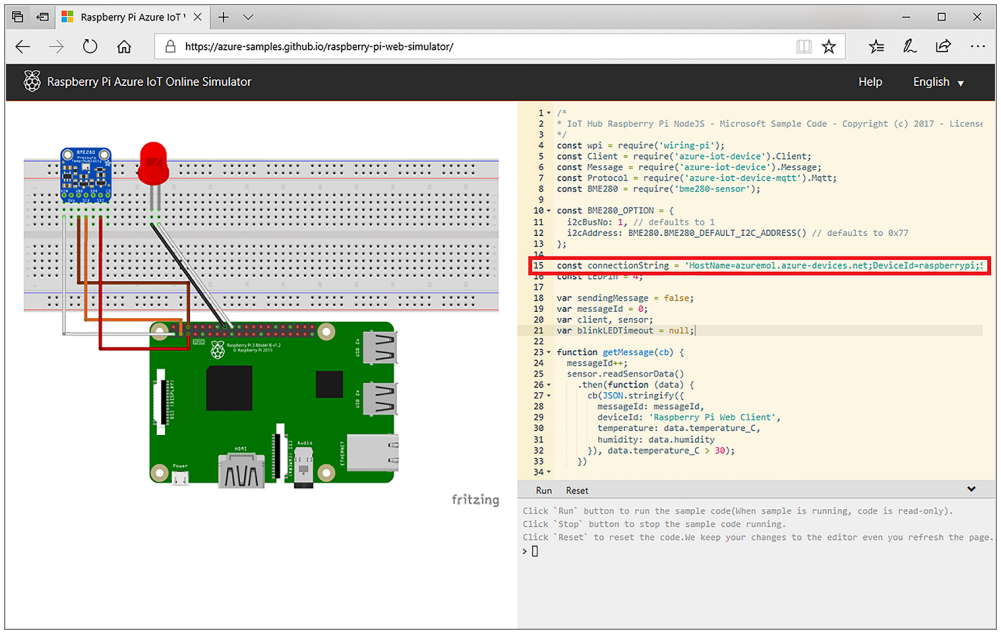


Рис. 20.3. Скопируйте и вставьте строку подключения устройства Интернета вещей Azure в симулятор Raspberry Pi. Переменная `connectionString` используется для подключения с целью передачи смоделированных данных датчика в Azure.

окне симулятора, можно увидеть, что в области строки 99 приложение определяет датчик. Симулятор имитирует действия реального датчика и генерирует данные, возвращаемые датчиком в приложение. Это базовый пример, поэтому подумайте, что еще можно было бы найти здесь: число оборотов в минуту (об/мин) двигателя или GPS-координаты транспортного контейнера либо грузовика и т. д. Здесь есть баланс между программной имитацией устройства и созданием функционального приложения с использованием реального оборудования и данных датчиков. Если вы хотите подробнее ознакомиться с Интернетом вещей Azure, в какой-то момент вам необходимо приобрести оборудование или взять его у кого-то во временное пользование.

6 Чтобы убедиться, что сообщения моделируемого устройства принимаются центром Интернета вещей, проверьте состояние квоты. Укажите имя центра Интернета вещей, например azuremol:

```
az iot hub show-quota-metrics --name azuremol
```

Выходные данные подобны представленному ниже примеру, в котором показывается, что получено 5 сообщений из максимально возможных 8 000 сообщений в день, и что есть одно подключенное устройство из максимально 500 возможных. Для получения этих показателей может потребоваться несколько минут, поэтому не беспокойтесь, если вы сразу не увидите никаких данных:

```
[
  {
    "currentValue": 5,
    "maxValue": 8000,
```

```
    "name": "TotalMessages"
  },
  {
    "currentValue": 1,
    "maxValue": 500,
    "name": "TotalDeviceCount"
  }
]
```

Также можете воспользоваться порталом Azure: выберите группу ресурсов, а затем — центр Интернета вещей. На странице «Обзор» на экране использования отображается количество полученных сообщений и подключенных устройств. Здесь тоже до появления сообщений и их записи в соответствии с квотой может пройти одна или две минуты. Полученные сообщения могут сразу же использоваться любыми приложениями, как мы увидим в разделе 20.4.

Беда в раю

Если вы не получаете сообщений в центре Интернета вещей, проверьте содержимое окна выходных данных моделируемого устройства Raspberry Pi. Одно из первых действий приложения — подключение к центру Интернета вещей Azure. Если строка подключения неверна, отображается сообщение об ошибке подключения. Проверьте, правильно ли была скопирована и вставлена вся строка подключения. Строка подключения начинается с `HostName`, а последним символом в каждом ключе доступа всегда является знак равенства (=).

Если в окне выходных данных отображается сообщение об ошибке, скопируйте текст этого сообщения в свою любимую поисковую систему и найдите соответствующий результат. Убедитесь, что вы не изменили другие строки кода, так как это может вызвать проблему! В окне кода нужно изменить только строку подключения.

Поскольку моделируемое устройство Raspberry Pi работает в веб-браузере, у вас может быть общая проблема с веб-сайтом. Попробуйте обновить страницу или получить доступ к симулятору в другом браузере (https://azure-samples.github.io/raspberry-pi-web-simulator).

20.4 Потоковая передача данных центра Интернета вещей Azure в веб-приложения Azure

Устройство, которое подключается к центру Интернета вещей, бесполезно, если вы ничего не можете сделать с данными. Здесь вы можете начать интегрировать многие сервисы и функции, о которых узнали в этой книге. Хотите выполнить потоковую передачу данных в таблицы или очереди службы хранилища Azure? Вы можете сделать это. Хотите обрабатывать данные от устройств Интернета вещей в виртуальных машинах или контейнерах Azure? Пожалуйста! Хотите использовать Azure Cosmos DB для репликации своих данных, а затем получать доступ к ним с помощью глобально избыточных веб-приложений и диспетчера трафика Azure? Нет проблем!

В рассматриваемом в качестве примера сценарии центр Интернета вещей является механизмом подключения и точкой входа в Azure для устройств Интернета вещей. Сам центр непосредственно ничего не делает с данными. Для событий существует конечная точка по умолчанию, представляющая собой большой контейнер для всех сообщений, получаемых от устройства Интернета вещей. Моделируемое устройство Raspberry Pi передает сообщения в центр Интернета вещей, который направляет их в эту конечную точку событий. Поток сообщений от устройств через центр Интернета вещей в конечную точку показан на рис. 20.4.

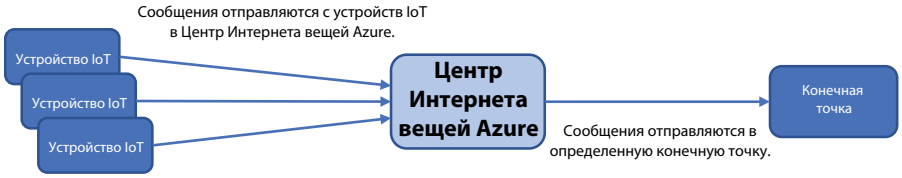


Рис. 20.4. Центр Интернета вещей получает сообщения от подключенных устройств Интернета вещей и отправляет их в конечную точку. Эти конечные точки могут использоваться другими сервисами Azure для получения данных от устройств Интернета вещей. Для событий существует конечная точка по умолчанию, из которой сервисы (веб-приложения) могут осуществлять чтение.

Можно создавать настраиваемые конечные точки, которые направляют сообщения непосредственно в сервисы Azure, например в хранилище и сервисную шину. В главе 4 были рассмотрены очереди хранилища Azure для передачи сообщений между приложениями. Более надежная и масштабируемая корпоративная платформа обмена сообщениями — служебная шина Azure. Сообщения можно добавлять в сервисную шину (например, данные, получаемые от устройств Интернета вещей), после чего другие приложения могут прослушивать эти сообщения и отвечать соответствующим образом.

Если усложнять решение чтением сообщений из очереди сообщений (например, сервисной шины) не требуется, можно использовать группы потребителей с конечной точкой событий по умолчанию. Группа потребителей позволяет сервисам, например веб-приложениям Azure, считывать данные с конечной точки (см. рис. 20.5). У каждого сервиса, получающего данные из центра Интернета вещей Azure, должна быть собственная группа потребителей. Несколько сервисов (каждый с собственной группой потребителей) могут получать одни и те же сообщения и обрабатывать их по мере необходимости.

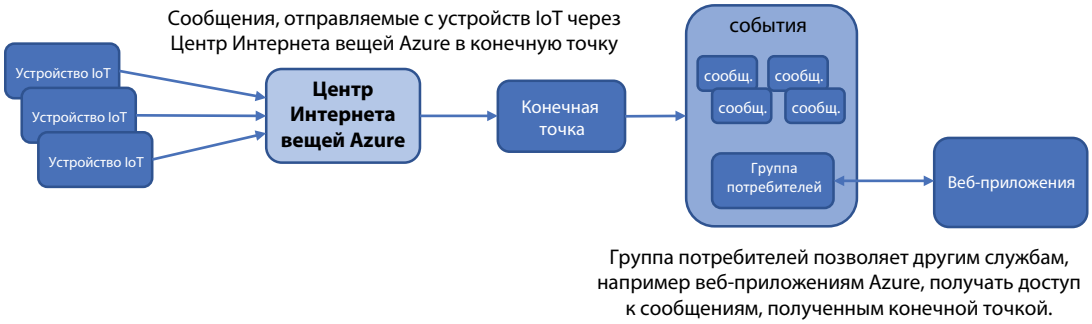


Рис. 20.5. Сообщения отправляются из устройств Интернета вещей в центр Интернета вещей, который затем направляет их в конечную точку. В каждой конечной точке можно создать группы потребителей. Они позволяют другим службам Azure получать доступ к сообщениям устройств, к которым иначе они не имели бы доступа. При наличии групп потребителей внешние приложения могут считывать данные устройств Интернета вещей без использования очереди сообщений.

Давайте создадим веб-приложение Azure, использующее группу потребителей для считывания данных сообщений в режиме реального времени с моделируемого устройства Raspberry Pi. В этом базовом примере показывается, как можно передавать данные от устройств Интернета вещей и получать к ним доступ из веб-приложений.

Попробуйте сейчас

Чтобы создать веб-приложение Azure, которое считывает данные с устройств Интернета вещей, выполните следующие действия:

1 Создайте в Cloud Shell план Azure App Service для веб-приложения. Введите имя, например `azuremol`. Для этих упражнений уровень «Бесплатный» (f1) достаточно хорош и снижает затраты:

```
az appservice plan create \
  --resource-group azuremolchapter20 \
  --name azuremol \
  --sku f1
```

2 Создайте веб-приложение. Укажите имя, например molwebapp, и разрешите использовать его с Git, чтобы можно было развернуть пример приложения. Как и в случае с другими публичными ресурсами Azure, требуется указать собственное глобально уникальное имя.

```
az webapp create \
  --resource-group azuremolchapter20 \
  --plan azuremol \
  --name molwebapp \
  --deployment-local-git
```

3 Для центра Интернета вещей определите группу потребителей, а также некоторые параметры приложения для веб-приложения. Эти параметры позволяют веб-приложению подключаться к центру Интернета вещей. На рис. 20.6 показано, что будет создано при выполнении нескольких следующих шагов.

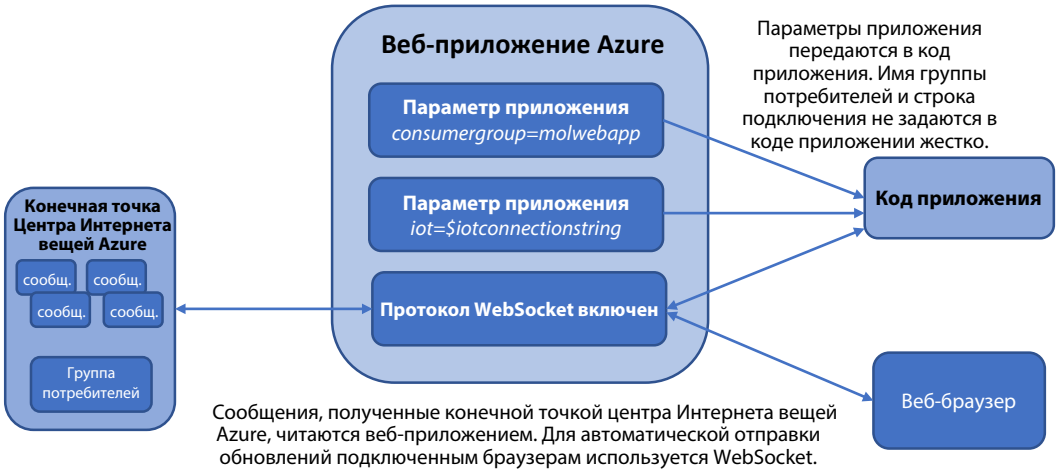


Рис. 20.6. Чтобы веб-приложение могло считывать данные с моделируемого устройства Raspberry Pi Интернета вещей, необходимо создать в центре Интернета вещей группу потребителей. Затем нужно определить для веб-приложения 2 параметра приложения, которые позволяют подключиться к группе потребителей. Чтобы веб-браузер автоматически получал поток данных от устройства Raspberry Pi по мере приема новых данных, также требуется включить параметр для WebSockets.

4 Создайте группу потребителей, которая позволяет веб-приложению получать доступ к данным о событиях, передаваемым от устройства Интернета вещей. Укажите центр Интернета вещей, например `azuremol`, и введите имя группы потребителей, например `molwebapp`. Обязательно используйте собственное имя на всех следующих шагах. Группа потребителей создается в конечной точке событий по умолчанию:

```
az iot hub consumer-group create \
  --hub-name azuremol \
  --name molwebapp
```

5 Веб-приложению необходимо сообщить, как называется группа потребителей. Создайте для веб-приложения параметр приложения, который используется в примере приложения, развертываемого в конце данного упражнения. Параметры приложения в веб-приложениях позволяют задавать определенные атрибуты, например имя группы потребителей и строку подключения, без жестко запрограммированных значений в приложении.

Укажите имя группы потребителей, созданной на шаге 4, например `mol webapp`:

```
az webapp config appsettings set \
  --resource-group azuremolchapter20 \
  --name molwebapp \
  --settings consumergroup=molwebapp
```

6 Чтобы подключиться к центру Интернета вещей, веб-приложению необходимо знать строку подключения для этого центра. Эта строка подключения отличается от той, которую вы скопировали для моделируемого устройства Raspberry Pi в предыдущем упражнении. Как указывалось ранее, для центра Интернета вещей существует строка подключения, которая использует политики общего доступа для определения разрешений на доступ. Строка подключения существует для каждого устройства Интернета вещей. Веб-приложению необходимо считывать данные из группы потребителей конечной точки центра Интернета вещей, поэтому следует определить строку подключения для самого центра Интернета вещей.

7 Получите строку подключения к центру Интернета вещей и назначьте ее переменной с именем `iotconnectionstring`, которая используется на шаге 8:

```
iotconnectionstring=$(az iot hub show-connection-string \
  --hub-name azuremol \
  --output tsv)
```

8 Создайте другой параметр приложения для веб-приложения, на этот раз для строки подключения центра Интернета вещей. Переменная, определенная на предыдущем шаге, позволяет приложению, используемому в качестве примера, подключаться к устройству Интернета вещей и считывать с него данные:

```
az webapp config appsettings set \
  --resource-group azuremolchapter20 \
  --name molwebapp \
  --settings iot=$iotconnectionstring
```

9 Включите протокол WebSocket. *WebSocket* — это средство двусторонней связи между браузером и сервером. Приложение, используемое в качестве примера, автоматически обновляет веб-браузер данными, полученными от устройства Raspberry Pi. Для выполнения этого автоматического обновления приложение использует WebSockets. Сервер может затем отправить данные в браузер и автоматически обновить его:

```
az webapp config set \
--resource-group azuremolchapter20 \
--name molwebapp \
--web-sockets-enabled
```

Давайте сделаем здесь паузу и обсудим, что было сделано к этому моменту. Вы работали с веб-приложениями во многих предыдущих главах, но с параметрами приложения для веб-приложений и средством WebSockets еще не встречались. На рис. 20.7 еще раз показано, как связаны веб-приложение и центр Интернета вещей.

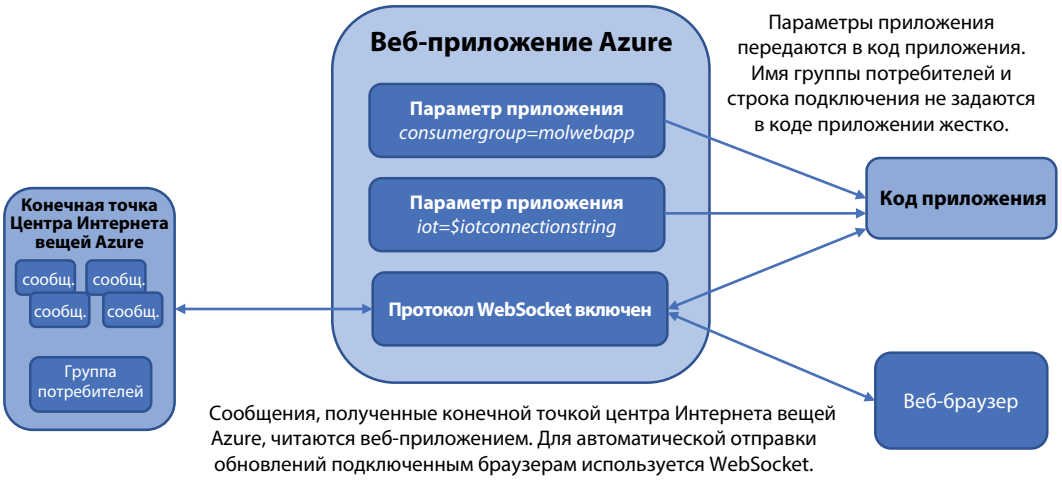


Рис. 20.7. Когда сообщения отправляются с устройств Интернета вещей, они проходят через центр Интернета вещей и направляются в конечную точку. Код приложения считывает параметры приложения веб-приложения, определяющие строку подключения центра Интернета вещей и группу потребителей, которую следует использовать. После подключения приложения к центру Интернета вещей группа потребителей позволяет веб-приложениям считывать сообщения устройств Интернета вещей. При каждом получении нового сообщения от устройства Интернета вещей веб-приложение использует соединение WebSocket с веб-браузерами, которые обращаются к вашему сайту для автоматической загрузки обновлений. Это соединение позволяет просматривать передаваемые от устройств Интернета вещей данные в режиме реального времени, например информацию о температуре и влажности от моделируемого устройства Raspberry Pi.

Теперь давайте завершим упражнение и развернем пример приложения из репозитория GitHub в ваше веб-приложение. После этого вы сможете открыть веб-приложение в браузере и просматривать в режиме реального времени данные, передаваемые от моделируемого устройства Raspberry Pi!

10 При необходимости скопируйте репозиторий примеров GitHub в Cloud Shell следующим образом:

```
git clone https://github.com/foulds/azure-mol-samples-2nd-ed.git
```

11 Перейдите в каталог для главы 20:

```
cd azure-mol-samples-2nd-ed/20
```

12 Инициализируйте репозиторий Git и добавьте основную веб-страницу:

```
git init && git add . && git commit -m "Pizza"
```

13 Чтобы загрузить пример приложения, создайте подключение к веб-приложению. Следующая команда получает репозиторий веб-приложения и настраивает репозиторий Git с локальными примерами для подключения к нему:

```
git remote add molwebapp \
  $(az webapp deployment source config-local-git \
  --resource-group azuremolchapter20 \
  --name molwebapp \
  --output tsv)
```

В предыдущих главах я просил вас заняться этой темой. Однако сейчас я надеюсь, что вы уже начали изучать возможности интерфейса командной строки Azure и поняли, что большую часть этой информации можно получить быстро:

14 Вставьте HTML-пример сайта в веб-приложение с помощью следующей команды:

```
git push molwebapp master
```

15 При появлении запроса введите пароль для пользователя Git, которого вы создали и использовали в предыдущих главах (эта учетная запись была создана в главе 3).

Если вы не записали пароль Git на стикере

Если вы забыли пароль, его можно сбросить. Сначала получите имя пользователя локальной учетной записи развертывания Git:

```
az webapp deployment user show --query publishingUserName
```

Для сброса пароля введите имя учетной записи из предыдущей команды, а затем ответьте на запросы, чтобы задать новый пароль. В следующем примере сбрасывается пароль для учетной записи пользователя с именем azuremol:

```
az webapp deployment user set --user-name azuremol
```

16 Просмотрите имя хоста веб-приложения, а затем откройте адрес в веб-браузере:

```
az webapp show \
  --resource-group azuremolchapter20 \
  --name molwebapp \
  --query defaultHostName \
  --output tsv
```

Для первого открытия сайта в веб-браузере может потребоваться несколько секунд, так как веб-приложение подключается к центру Интернета вещей, запускает подключение WebSocket и ожидает получения первого сообщения устройства. Каждые 2 секунды веб-браузер должен автоматически обновляться последними смоделированными данными от устройства Raspberry Pi (см. рис. 20.8).

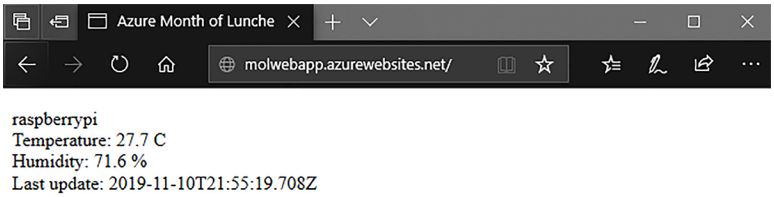


Рис. 20.8. В примере приложения используется соединение WebSocket между веб-браузером и веб-приложением для автоматического обновления каждые 2 секунды последними данными от моделируемого устройства Raspberry Pi.

Если в экземпляре веб-приложения не отображаются никакие данные, убедитесь, что моделируемое устройство Raspberry Pi по-прежнему работает. При необходимости запустите моделируемое устройство и убедитесь, что оно подключается к Интернету вещей Azure и отправляет сообщения. После этого данные должны начать появляться в экземпляре веб-приложения.

20.5 Обзор компонентов Интернета вещей Azure

Я надеюсь, что упражнения в этой главе дали вам представление о том, какие сервисы есть в Azure для решений Интернета вещей:

- *Центр Интернета вещей Azure* предоставляет отличные возможности для подготовки, подключения множества устройств Интернета вещей, управления ими и их последующей интеграции с другими сервисами Azure.
- *Акселераторы решений Интернета вещей Azure* предоставляют встроенные сценарии, которые автоматически объединяют множество сервисов Azure для создания полнофункциональной среды приложений.
- *Azure IoT Edge* позволяет развернуть сервисы Azure в локальной среде для обработки данных от устройств Интернета вещей без централизованной потоковой передачи всех данных в Azure.

Чтобы подробнее изучить Интернет вещей Azure и устройства Интернета вещей в целом, я рекомендую приобрести базовое устройство Raspberry Pi или аналогичное устройство. Эти устройства относительно дешевы, часто поставляются с несколькими различными базовыми датчиками или электрическими компонентами для проверки различных идей и предоставляют отличную платформу для обучения, поскольку позволяют увидеть, что возможно при интеграции оборудования и программного обеспечения. Только не забудьте предупреждения из главы 17 об искусственном интеллекте, машинном обучении и создании Skynet! В издательстве Manning также есть отличные книги, например *Building the Web of Things* Доминика Гинара (Dominique D. Guinard) и Влада Трифа (Vlad M. Trifa) (https://www.manning.com/books/building-the-web-of-things) и *JavaScript on Things* Лайзы Дэнджер Гарднер (Lyza Danger Gardner) (https://www.manning.com/books/javascript-on-things), в которых подробно рассматриваются устройство Raspberry Pi, лучшие способы использования Интернета вещей, а также программирование на JavaScript и Node.js для Интернета вещей.

Помните об упомянутой мною необходимости всегда удалять группы ресурсов?

В этой книге рекомендуется удалять группы ресурсов по окончании изучения каждой главы. Такой подход гарантирует, что платные сервисы и приложения не будут продолжать работать, когда они вам не нужны.

(продолжение)

Интернет вещей Azure предоставляет отличную платформу для потоковой передачи данных в Azure. Эти данные обычно требуется обрабатывать, а не просто отображать в веб-приложении, как это делалось в упражнениях. В главе 21 рассматриваются бессерверные вычисления с помощью сервисов Logic Apps и «Функции».

Чтобы увидеть, насколько хорошо эти сервисы Azure работают вместе, не удаляйте группу ресурсов и сервисы, развернутые в этой главе. Вы будете использовать их сразу же в начале главы 21, чтобы узнать, как можно предпринимать действия на основе данных, полученных от устройств Интернета вещей. Просто вернитесь к своей имитации устройства Raspberry Pi и нажмите кнопку Stop, иначе лимит в 8000 сообщений будет исчерпан довольно быстро!

20.6 Практическое упражнение: изучение вариантов использования Интернета вещей

В этой главе обсуждалось много нового, но без реального устройства Интернета вещей вы ограничены в своих возможностях. В главе 21 используется центр Интернета вещей Azure и имитация устройства Raspberry Pi, поэтому сейчас я не хочу выполнять слишком много настроек. Ниже приведены несколько вопросов и предложений для дальнейшего изучения вами Интернета вещей.

1. Как вы думаете, в каких областях устройства Интернета вещей могли бы принести пользу вашему бизнесу? Если вы сейчас не занимаетесь бизнесом, воспользуйтесь вымышленным *магазином пиццы* Azure Month of Lunches.
2. Что вы можете сделать с помощью Интернета вещей, чтобы улучшить обслуживание клиентов?
3. Будете ли вы использовать Azure IoT Edge? Почему?
4. Какие другие сервисы Azure вы, скорее всего, интегрируете для запуска своих приложений?
5. Если у вас есть время, попробуйте воспользоваться одним из акселераторов решений Интернета вещей Azure, предлагаемых по адресу www.azureiotsolutions.com/Accelerators. Существует сценарий моделирования устройств, в котором создается виртуальная машина и моделируются датчики. Этот сценарий похож на имитацию устройства Raspberry Pi, но предоставляет гораздо больше возможностей! Для подготовки всех необходимых ресурсов потребуется несколько минут. Затем посмотрите на портал Azure, чтобы увидеть, что было создано, и совместную работу всех компонентов.
6. Видите, как используются сервисы из предыдущих глав, в частности сервис хранилища и Cosmos DB?
7. Какие еще есть акселераторы решений Интернета вещей? Совпадают ли какие-либо из них с идеями, которые у вас были для ваших собственных приложений?

21 Бессерверные вычисления

В этой заключительной главе давайте заглянем в будущее с бессерверными вычислениями. Если вы разработчик, идея контейнеров, рассмотренная в главе 19, может быть для вас привлекательной из-за отсутствия необходимости настраивать базовую инфраструктуру для приложений. Если это так, вам понравятся компоненты Azure для бессерверных приложений! Если же вы ИТ-администратор, перед которым вдруг встал вопрос, в чем будет заключаться ваша работа, если в будущем не будет серверов, не беспокойтесь! *Бессерверные вычисления* — это скорее маркетинговый термин, и вы будете продолжать применять многие навыки работы с серверами и инфраструктурой!

В Azure бессерверные вычислительные возможности предоставляются двумя основными предложениями — Azure Logic Apps и приложениями-функциями Azure. В этой главе мы рассмотрим, что предлагает каждый из этих сервисов и как они работают вместе. Чтобы бессерверные приложения могли взаимодействовать друг с другом и передавать данные, требуются сервисы обмена сообщениями. Поэтому мы также рассмотрим такие сервисы, в частности сетку событий Azure, сервисную шину и концентраторы событий.

21.1 Что такое бессерверные вычисления?

Сказать, что бессерверные вычисления — это просто вычисления без сервера, неправильно, так как сервер где-то выполняет для вас некоторый код. Отличие от рабочих нагрузок приложений IaaS, в частности виртуальных машин Azure и рабочих нагрузок PaaS в веб-приложениях, заключается в том, что бессерверные приложения обычно разбиваются на мелкие отдельные блоки. Вы запускаете не одно большое приложение, а его небольшие компоненты. Если это похоже на контейнеры и микросервисы, которые были рассмотрены в главе 19, не думайте, что вы запутались, так как бессерверные вычисления во многом совпадают с этими темами с точки зрения разработки приложений. Вы можете создавать микросервисы, используя бессерверные вычисления, что мы рассмотрим в этой главе.

На рис. 21.1 показано разбиение приложения на небольшие компоненты, которые работают в сервисе бессерверных вычислений и предоставляют небольшие блоки выходных данных.

Рис. 21.1. В бессерверной вычислительной среде каждое приложение разбивается на небольшие дискретные блоки компонентов. Каждый компонент работает в сервисе бессерверных вычислений, например в сервисе приложений-функций Azure, и создает выходные данные, которые затем могут использоваться другими компонентами бессерверных приложений или другими сервисами Azure, например Azure IoT или хранилищем Azure.

В Azure бессерверные вычисления охватывают два указанных ниже основных сервиса

- *Azure Logic Apps*. Чтобы реагировать на определенные входные данные и триггеры, приложения логики позволяют визуально создавать рабочие процессы, которые могут обрабатывать и инициировать дополнительные действия простым и понятным способом, не требующим написания кода. Приложения логики могут создаваться пользователями, не имеющими опыта программирования и работы с ИТ-инфраструктурой. На рис. 21.2 показана простая блок-схема приложения логики.

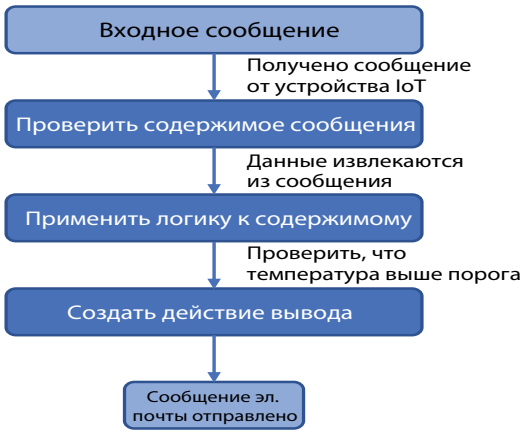


Рис. 21.2. В приложении логики входные данные могут появляться при публикации твита, загрузке файла или получении сообщения от устройства Интернета вещей. Приложение логики применяет для этих данных правила и фильтры и определяет, удовлетворяет ли сообщение определенным вами критериям. Затем выполняются выходные действия, например создается сообщение электронной почты. Вся эта логика не требует программирования и инфраструктуры приложений, кроме подписки Azure.

  Не требуется поддержка обновлений системы безопасности и нет проектных требований к высокой доступности и возможности масштабирования. Платформа Azure выполняет эти функции автоматически. Для приложений логики существуют сотни встроенных коннекторов, предназначенных для интеграции с такими сервисами, как Twitter, Office 365, SharePoint и Outlook. Вы можете отвечать на общедоступные твиты о вашей компании или продукте, отправлять оповещение по электронной почте при загрузке файла в SharePoint или передавать уведомление при получении сообщения от устройства Интернета вещей.
- *Приложения-функции Azure*. Для запуска небольших блоков кода приложения-функции позволяют использовать широко распространенные языки программирования, например C#, Node.js или Python, без дополнительного

управления инфраструктурой. Код выполняется в безопасной изолированной среде, и вам выставляется счет в зависимости от использования памяти в секунду. На рис. 21.3 представлена базовая блок-схема для приложения-функции.

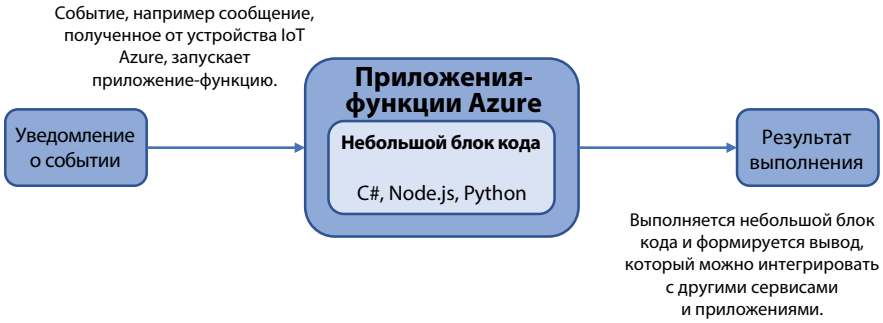


Рис. 21.3. Как и в случае с приложением логики, уведомление о событии (триггер события) обычно запускает функцию Azure. Приложение-функция содержит небольшой блок кода, который выполняет конкретную задачу. Нет инфраструктуры для настройки и обслуживания. Требуется только небольшой блок кода. После завершения выполнения кода выходные данные могут быть интегрированы с другим сервисом или приложением Azure.

Нет виртуальных машин для обслуживания, и не требуются веб-приложения. Вам не нужно думать об обеспечении высокой доступности и масштабирования, так как эти задачи выполняет сервис приложений-функций Azure. Вам необходимо предоставить только код. Платформа Azure гарантирует доступность ресурсов для выполнения этого кода в любой момент, когда вам потребуется его запустить.

Приложениям логики код не требуется, поэтому у этих приложений более широкая потенциальная база пользователей. Например, владельцы бизнес-приложений или группы по финансам и бухгалтерскому учету могут создавать собственные приложения логики без написания кода. Приложения-функции обеспечивают больше контроля и гибкости, а также позволяют обрабатывать события определенным образом и лучше интегрировать их с другими компонентами приложения.

Как приложения логики, так и приложения-функции позволяют выполнять действия на основе триггеров без необходимости поддерживать среду или инфраструктуру приложений. Приложение логики или приложение-функция выполняется сервером, расположенным в Azure, но с точки зрения ИТ-администратора или разработчика это бессерверные технологии.

21.2 Платформы обмена сообщениями Azure

В главе 12 было рассмотрено отслеживание ресурсов Azure и устранение неполадок с ними, а в главе 16 — использование центра безопасности Azure для обнаружения проблем и управления обновлениями. Обе функции используют потоки данных, например расширение диагностики виртуальной машины Azure, для информирования платформы о том, что происходит в виртуальной машине. Платформы диагностики и мониторинга Azure прекрасны, а другие сервисы, например веб-приложения, экземпляры контейнеров Azure и центр Интернета вещей Azure, также могут выполнять потоковую передачу диагностических данных служб для централизованного анализа.

При использовании бессерверных приложений часто требуется осуществлять обмен сообщениями и передавать фактические данные приложения, а не только диагностические данные устранения неполадок и данные обновления состояний. Вот когда вам нужна платформа обмена сообщениями.

21.2.1 Сетка событий Azure

Что делать, если вам просто требуется сообщать об определенных выполняемых действиях или операциях? В рабочих процессах автоматизации и бессерверных вычислениях полезно иметь возможность выполнения действий в ответ на события (см. рис. 21.4).

Рис. 21.4. Сервисы Azure, такие как Azure IoT и хранилище Azure, могут отправлять уведомления в сетку событий Azure. Эти уведомления могут появляться при получении сообщения от устройства Интернета вещей или при загрузке файла в хранилище. Сетка событий Azure позволяет другим сервисам и поставщикам подписываться на эти уведомления для выполнения дополнительных действий в ответ на события.

Рассмотрим пару сценариев, которые вы можете использовать в своем магазине пиццы:

- *В центре Интернета вещей получено сообщение*. Устройство Интернета вещей, подключенное к центру Интернета вещей, может сообщить о температуре в духовке или о местоположении транспортного средства доставки. Центр Интернета вещей настроен для пересылки уведомлений в сетку событий Azure.

  Функция Azure подписана на уведомления сетки событий для центра Интернета вещей и запускает небольшой компонент бессерверного приложения для регистрации информации в Cosmos DB и отправки уведомления по электронной почте. Вместо приложений-функций Azure можно также использовать приложения логики в зависимости от того, насколько сложным должен быть ответ приложения.
- *В хранилище Azure загружен файл*. Отдел маркетинга может загрузить в хранилище купон на скидку, чтобы сэкономить деньги при заказе пиццы. При создании нового файла в сетку событий отправляется уведомление.

  Веб-перехватчик подписан на сетку событий и публикует копию изображения из хранилища в Twitter. Этот твит позволяет клиентам узнать о сделке недели или купоне, позволяющем сэкономить деньги.

Эти сценарии предназначены для автоматических бессерверных вычислительных сценариев, но сетка событий также может интегрироваться с традиционными ресурсами — виртуальными машинами и веб-приложениями. Например, группу ресурсов можно

настроить для отправки уведомлений в сетку событий. Существует множество способов создания виртуальной машины, например на портале, с помощью интерфейса командной строки Azure или шаблона диспетчера ресурсов, поэтому необходимо убедиться в правильности настройки виртуальной машины для управления обновлениями через центр обеспечения безопасности. Модуль Runbook сервиса автоматизации Azure может быть подписан на сетку событий для уведомлений об операциях создания виртуальных машин. Затем он может размещать виртуальные машины в сервисе управления обновлениями и устанавливать необходимые обновления безопасности или приложения.

21.2.2 Концентраторы событий и сервисная шина Azure

Сетка событий может работать со многими ресурсами Azure и хорошо подходит для бессерверных вычислений с помощью приложений логики или приложений-функций. Однако приложения логики и приложения-функции могут работать на основе других информационных входов, например концентраторов событий или сервисной шины. Давайте рассмотрим различия между этими сервисами обмена сообщениями, чтобы вы могли принять оптимальное решение в отношении их использования.

- *Концентраторы событий Azure* позволяют получать поток данных, например от устройств Интернета вещей или системы дистанционного отслеживания приложений. Они предоставляют платформу обмена сообщениями с низкой задержкой, способную обрабатывать миллионы событий в секунду от нескольких параллельных поставщиков. Концентраторы событий — это хранилище данных, а не очередь сообщений. При этом клиент или приложение проверяет наличие событий в концентраторе с любой частотой. Затем данные, полученные в концентраторе событий, могут быть обработаны другими сервисами, как показано на рисунке 21.5.
- *Сервисная шина Azure* позволяет компонентам приложения обмениваться такими данными сообщений, как очереди хранилища, которые были рассмотрены в главе 4. Очереди хранилища — это ранняя, базовая реализация платформы обмена сообщениями в Azure. *Сервисная шина* поддерживает расширенные функции,

Рис. 21.5. Устройства Интернета вещей подключаются к центру Интернета вещей и могут передавать все данные своих датчиков. Могут быть подключены сотни или тысячи устройств Интернета вещей. Концентраторы событий Azure обрабатывают все эти отдельные потоки данных и позволяют таким сервисам, как Azure HDInsight, обрабатывать необработанные данные в кластерах Hadoop или Spark для анализа и создания отчетов.

например обеспечение гарантированного порядка сообщений, атомарные операции и отправку сообщений партиями. На рис. 21.6 представлен общий сценарий для сервисной шины.

Рис. 21.6. Сообщения помещаются в очередь сервисной шины компонентами приложения, в данном примере — клиентским приложением. Затем сообщения могут извлекаться и обрабатываться промежуточными или серверными приложениями по мере необходимости. На данном рисунке сообщение извлекается и обрабатывается серверным приложением. Расширенные функции обмена сообщениями включают обеспечение гарантированного порядка сообщений в очереди, блокировку сообщений, тайм-ауты и ретрансляцию.

Какой из 3 сервисов, позволяющих передавать, получать и обрабатывать данные между приложениями и сервисами в Azure, следует использовать и когда? В табл. 21.1 приведены общие сведения о сервисах «Сетка событий», «Концентраторы событий» и «Сервисная шина».

Таблица 21.1. Каждый сервис предназначен для определенного сценария. Сетка событий позволяет реагировать на события, концентраторы событий служат для потоковой передачи больших объемов данных, а сервисная шина предназначена для передачи сообщений между сервисами и компонентами приложений.

Сервис Azure	Назначение	Сценарий использования
Сетка событий	Распределение событий	Выполнение дополнительного действия в зависимости от события.
Концентраторы событий	Потоки данных	Получение и передача больших объемов параллельных данных.
Служебная шина	Передача сообщений	Обеспечение связи между сервисами и приложениями.

Приложения логики и приложения-функции Azure могут запускаться всеми 3 платформами обмена сообщениями. Давайте создадим сервисную шину, которая может использоваться для запуска приложения логики.

21.2.3 Создание сервисной шины и ее интеграция с центром Интернета вещей

В этом сценарии сервисная шина используется для передачи сообщений, полученных из центра Интернета вещей. Моделируемое устройство Raspberry Pi из главы 20 генерирует показания температуры и передает их в центр Интернета вещей. Если температура превышает 30 °C, в сообщение от устройства Интернета вещей включается другой компонент данных: `temperatureAlert = true`. На рис. 21.7 показано, как интегрировать центр Интернета вещей с сервисной шиной для обработки сообщений с этим оповещением о температуре.

Рис. 21.7. Когда моделируемое устройство Интернета вещей Raspberry Pi отправляет данные сообщения, при показании температуры 30 °C и выше создается оповещение. Сообщения с этим оповещением направляются в сервисную шину. После этого их можно использовать для запуска приложений логики.

Попробуйте сейчас

Чтобы создать сервисную шину, выполните следующие действия:

1. Откройте портал Azure и в верхнем левом углу меню выберите «Создать ресурс».
2. Найдите и выберите сервисную шину, а затем нажмите «Создать».
3. Укажите имя, например `azuremol`, а затем выберите базовую ценовую категорию.
4. Выберите «Создать группу ресурсов» и укажите имя, например `azuremolchapter21`. Убедитесь, что местоположение такое же, как и для ресурсов, созданных в главе 20, — Восток США. Если местоположения не соответствуют, при взаимодействии между очередью сервисной шины, приложением логики и приложением-функцией могут возникнуть проблемы.
5. Примите другие значения по умолчанию и создайте сервисную шину.
6. После создания ресурса выберите группу ресурсов, а затем сервисную шину, созданную на шаге 5.
7. Выберите «Очереди», добавьте новую очередь и введите имя, например `azuremol`.
8. Оставьте все другие значения по умолчанию и нажмите «Создать».

Как при созданной сервисной шине и очереди настроить центр Интернета вещей для их использования? В центре Интернета вещей в качестве адресатов сообщений, получаемых от устройств Интернета вещей, определяются *конечные точки*. Для всех сообщений, которые не соответствуют заданным критериям, в центре Интернета вещей существует конечная точка по умолчанию. Сервисную шину можно настроить в качестве конечной точки получения сообщений. Затем определяется *маршрут*, включающий критерии, по которым сообщения должны направляться в конечную точку. В данном примере такой маршрут требует, чтобы любое сообщение, в тексте которого содержится `temperatureAlert = true`, направлялось в конечную точку «Сервисная шина» (см. рис. 21.8).

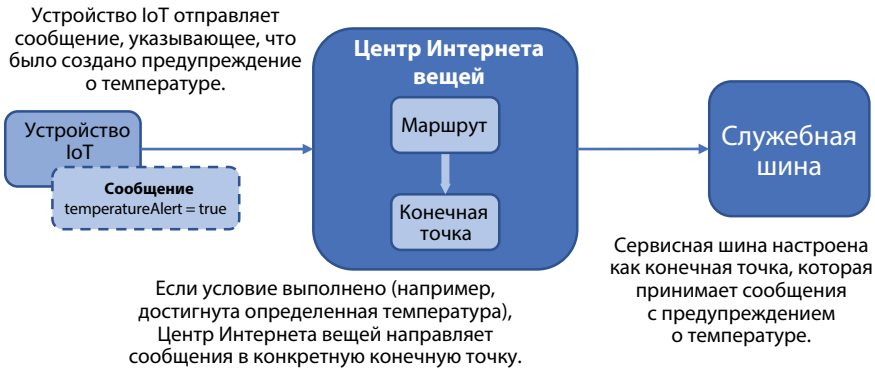


Рис. 21.8. Поскольку сообщения передаются от устройств Интернета вещей в центр Интернета вещей, их можно направлять в конкретные конечные точки в зависимости от определенных критериев. Сообщения, в тексте которых содержится оповещение о температуре, могут направляться в конечную точку, использующую очередь сервисной шины. Затем сообщения, помещенные в очередь сервисной шины и содержащие оповещение о температуре, могут использоваться для запуска приложений логики или приложений-функций Azure.

Попробуйте сейчас

Чтобы настроить центр Интернета вещей для направления сообщений с оповещениями о температуре в сервисную шину, выполните следующие действия:

1. Выберите группу ресурсов из главы 20, например `azuremolchapter20`, а затем центр Интернета вещей.
2. В разделе «Сообщения» на панели навигации слева выберите «Маршрутизация сообщений». Затем добавьте пользовательскую конечную точку для очереди сервисной шины.
3. Введите имя конечной точки, например `azuremol`.
4. Выберите пространство имен очереди сервисной шины, например `azuremol`, а затем свою фактическую очередь.
5. Чтобы направлять сообщения в эту конечную точку, создайте маршрут. В разделе «Маршрутизация сообщений» на панели навигации слева выберите «Маршруты» и добавьте новый маршрут.
6. Укажите имя, например `temperatureAlert`.
7. Выберите конечную точку сервисной шины, созданную на предыдущем шаге, например `azuremol`.
8. В качестве запроса маршрутизации введите следующую строку:

```
temperatureAlert = "true"
```

9. Когда все будет готово, сохраните маршрут.

Теперь у вас есть моделируемое устройство Raspberry Pi, которое отправляет данные в центр Интернета вещей, и маршрут для помещения сообщений с оповещением о температуре в очередь сообщений сервисной шины. Приложения у вас пока нет, поэтому вы ничего не можете сделать с данными в очереди сервисной шины. Что вы

можете сделать с оповещением о температуре? Обычный пример — отправка уведомления по электронной почте, поэтому давайте посмотрим, как запускать приложение логики каждый раз, когда сообщение помещается в очередь сервисной шины.

21.3 Создание приложения логики Azure

Как вы видели при рассмотрении приложений логики в разделе 21.1, сообщение, полученное из очереди сервисной шины, может использоваться в качестве триггера для запуска процесса выполнения. Для обработки сообщений, получаемых от устройств Интернета вещей, используется Центр Интернета вещей, и в конечную точку очереди сервисной шины направляются только сообщения, в тексте которых есть строка `temperatureAlert = true`. При таком подходе приложение логики запускается только при создании оповещения о температуре.

На рис. 21.9 показано, что делает приложение логики. Когда сообщение помещается в очередь сервисной шины, приложение логики запускается и отправляет оповещение по электронной почте.

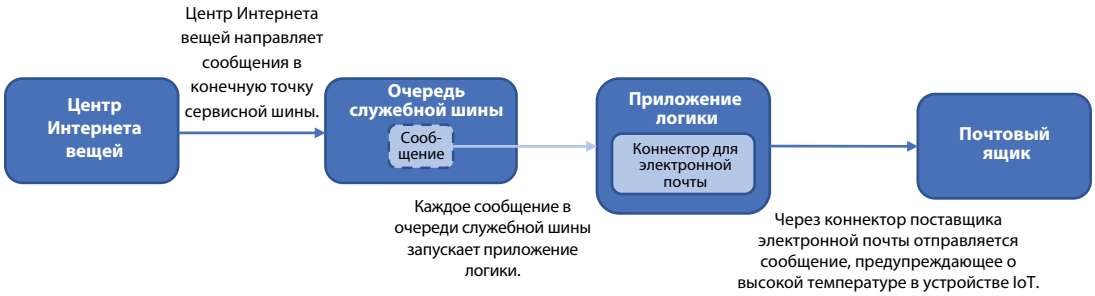


Рис. 21.9. При каждом поступлении сообщения из центра Интернета вещей в очередь сервисной шины запускается приложение логики. При этом оно отправляет уведомление по электронной почте через определенного поставщика почтовых услуг.

Попробуйте сейчас

Чтобы создать приложение логики, выполните указанные ниже действия:

1. На портале Azure выберите в меню вверху слева «Создать ресурс».
2. Найдите и выберите приложение логики, а затем нажмите «Создать».
3. Укажите имя, например `azuremol`, и выберите группу ресурсов, например `azuremolchapter21`. Опять же, выберите то же расположение, что и для других ресурсов IoT из главы 20.
4. Оставьте другие значения по умолчанию и нажмите «Создать».
5. После создания ресурса выберите группу ресурсов и откройте приложение логики. Для параметра «Добавить общие триггеры» выберите значение «При получении сообщения в очереди служебной шины».
6. Укажите имя, например `azuremol`, и выберите очередь сервисной шины, например azuremol.

7 Выберите политику сервисной шины по умолчанию, например RootManageSharedAccess-Key, и создайте подключение.

8 Нажмите «Продолжить», а затем выберите имя очереди сервисной шины, например `azuremol`.

9 Примите значения по умолчанию, такие как частота проверки сообщений.

10 Добавьте новый шаг в приложение логики.

11 Чтобы добавить действие, найдите то, что вы хотите выполнить. В этом упражнении выполните поиск по ключевым словам *электронная почта*. Выберите поставщика, например Gmail — Send an Email (Gmail — отправить электронное письмо), Outlook.com — Send an Email (Outlook.com — отправить электронное письмо) или SMTP — Send an Email (SMTP — отправить электронное письмо) (см. рис. 21.10).

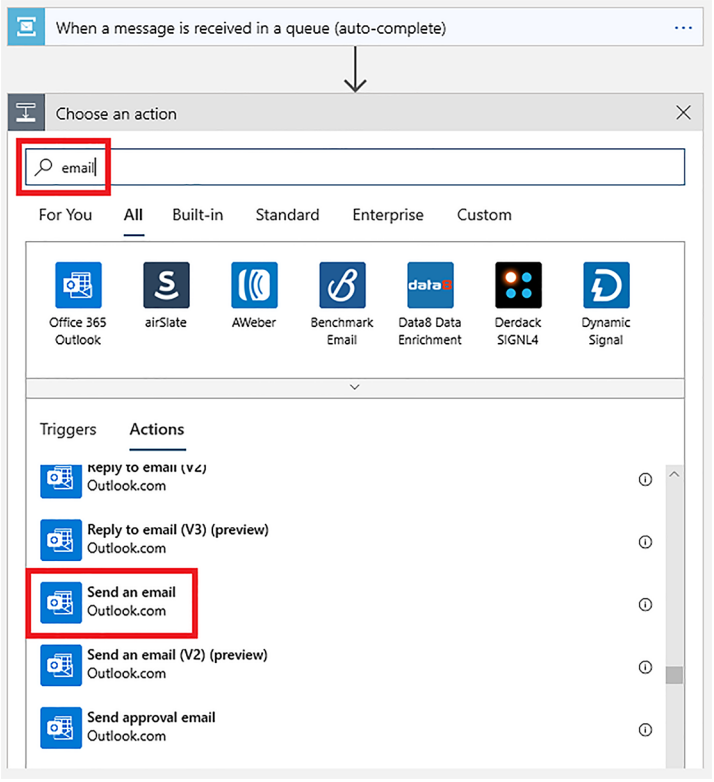


Рис. 21.10. Найдите и выберите текущего поставщика услуг электронной почты, например Gmail или Outlook.com. Можете также выбрать SMTP — Send an Email (SMTP — отправить электронное письмо), чтобы вручную настроить другого поставщика.

12 Войдите в систему поставщика электронной почты, чтобы разрешить маршрутизацию почты и подтвердить, что вы хотите предоставить приложениям логики разрешения на отправку электронной почты.

13 Введите адрес электронной почты получателя, по которому вы получаете электронную почту, тему сообщения, например Оповещение о температуре, и текст сообщения, например В устройстве Интернета вещей обнаружена высокая температура.

14 Сохраните приложение логики.

Давайте сделаем паузу и рассмотрим, что было создано в последних нескольких упражнениях (см. рис. 21.11). В этом базовом проекте бессерверного приложения нет элементов управления, ограничивающих количество отправляемых сообщений. В приложении логики можно определить, что требуется отправлять не больше пяти оповещений по электронной почте, а затем перед отправкой дополнительных сообщений нужно подождать 30 минут. В ходе разработки приложения вы должны подумать о том, как вы хотите получать уведомления о подобных ситуациях. Приложение логики можно, кроме того, настроить для чтения данных сообщения из очереди сервисной шины, а также включения временной метки сообщения устройства Интернета вещей и фактически зарегистрированной температуры. В следующем упражнении мы рассмотрим, как это сделать.

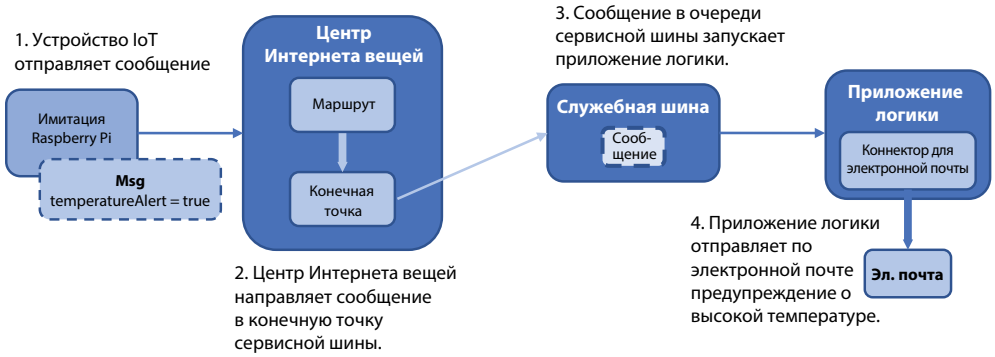


Рис. 21.11. Моделируемое устройство Raspberry Pi отправляет в центр Интернета вещей сообщение с показаниями датчика температуры каждые 2 секунды. Если температура превышает 30 °C, появляется оповещение о температуре. Центр Интернета вещей направляет все сообщения, содержащие оповещение о температуре, в очередь служебной шины. Сообщения в этой очереди вызывают запуск приложения логики Azure. Приложение логики подключается к поставщику услуг электронной почты, например Outlook или Gmail, и отправляет по электронной почте уведомление о получении от устройства Интернета вещей предупреждения о температуре.

Давайте рассмотрим это базовое бессерверное приложение в действии.

Попробуйте сейчас

Чтобы запустить моделируемое устройство Raspberry Pi и протестировать приложение логики, выполните следующие действия:

1 Откройте браузер для имитируемого устройства Raspberry Pi IoT из главы 20 (https://azure-samples.github.io/raspberry-pi-web-simulator).

2 Убедитесь, что строка подключения центра Интернета вещей по-прежнему присутствует в окне кода из главы 20.

3 Запустите приложение.

Моделируемые показания датчика температуры и влажности создаются каждые 2 секунды, и в центр Интернета вещей отправляется сообщение. До появления моделируемого показания температуры 30 °C и его отображения в окне выходных данных может быть отправлено несколько сообщений.

Центр Интернета вещей направляет все сообщения, содержащие строку `temperatureAlert: true`, в конечную точку, которой является сервисная шина. По мере того как эти сообщения помещаются в очередь сервисной шины, приложение логики выбирает их и отправляет через определенного поставщика сообщение электронной почты. Вы получаете по электронной почте сообщение, уведомляющее о высокой температуре. Этот процесс должен занять всего несколько секунд.

4 Моделируемое устройство Raspberry Pi генерирует сообщения каждые 2 секунды, поэтому остановите приложение, если не хотите получить по электронной почте много оповещений!

При получении оповещений по электронной почте сообщения не содержат большого количества информации. Приложение логики не извлекает содержимое сообщения из служебной шины и не форматирует информацию. Было бы замечательно, если бы оповещение электронной почты могло содержать имя устройства Интернета вещей или зарегистрированную температуру. Как можно обрабатывать каждое сообщение и выполнять некоторый анализ? Что можно сказать по поводу другого рассмотренного нами сервиса Azure для бессерверных приложений — приложений-функций Azure?

21.4 Создание приложения-функции Azure для анализа данных устройства Интернета вещей

Чтобы расширить имеющееся бессерверное приложение, можно запустить приложение-функцию Azure в приложении логики. Данные сообщения из сервисной шины могут быть отправлены в приложение-функцию для анализа зарегистрированной температуры. Уведомление по электронной почте, отправляемое приложением логики, может в этом случае содержать сведения об имени устройства Интернета вещей и зарегистрированной температуре. На рис. 21.12 показано взаимодействие между приложением логики и приложением-функцией.

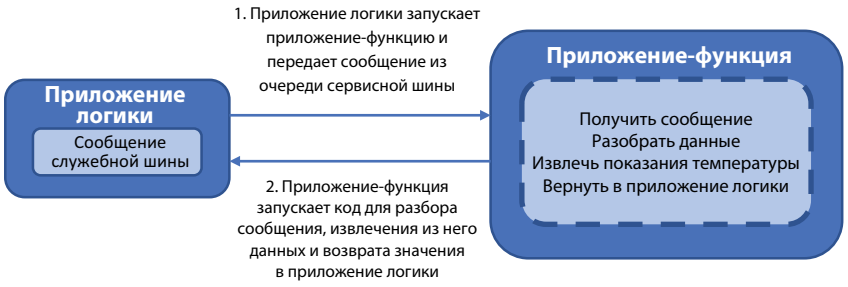


Рис. 21.12. Приложение логики запускает приложение-функцию. Сообщение, полученное в очереди сервисной шины, передается в функцию. Код в приложении-функции анализирует сообщение, извлекает значение температуры и возвращает его приложению логики. Для выполнения этого кода приложению-функции требуется несколько миллисекунд, поэтому затраты на выполнение этих вычислительных задач составляют доли цента.

Попробуйте сейчас

Чтобы создать приложение-функцию и запустить ее из приложения логики, выполните следующие действия:

1. На портале Azure выберите в меню вверху слева «Создать ресурс».
2. Найдите и выберите приложение-функцию, а затем нажмите «Создать».
3. Выберите группу ресурсов, например `azuremolchapter21`, и введите имя, например `azuremol`. Необходимо использовать тот же регион, что и для предыдущих ресурсов.

   Вам нужно опубликовать код, но обратите внимание, что также можно опубликовать образ контейнера Docker (глава 19). Вам даже не нужно создавать экземпляр контейнера или дополнительную инфраструктуру — недолговечный контейнер будет работать по мере необходимости, а затем останавливаться.
4. Для этого базового приложения выберите среду выполнения Node.js, так как мы используем простой JavaScript.
5. Вам доступны 3 параметра плана хостинга. *План потребления* позволяет платить за выполнение. При использовании этого плана требуемые ресурсы динамически назначаются во время выполнения. Для согласованных приложений, готовых к работе, можно использовать план *«Премиум»* или *выделенный план хостинга,* обеспечивающий фиксированную предсказуемую стоимость. Планы «Премиум» предоставляют дополнительные функции, такие как защита подключения к определенному набору виртуальных сетей Azure и гарантируют наличие экземпляра, позволяющего предотвратить задержки при холодном запуске приложения. Для этого упражнения выберите план потребления.
6. Примите другие значения по умолчанию для создания учетной записи хранения и Application Insights, а затем выберите «Проверить и создать».
7. Когда все будет готово, создайте приложение-функцию. Для этого потребуется 1–2 минуты.
8. После создания ресурса выберите группу ресурсов, откройте приложение логики из предыдущего упражнения и нажмите «Изменить».
9. В конструкторе приложений логики выберите «Добавить новый шаг».
10. Найдите и выберите «Функции Azure», а затем выберите функцию, созданную на предыдущих шагах (например, `azuremol`). Затем нажмите кнопку «Создать функцию».
11. Укажите имя функции, например `analyzeTemperature`.
12. Удалите существующий код, замените его кодом из следующих фрагментов кода, а затем нажмите «Создать».

    Этот код также доступен в репозитории GitHub по адресу web-address.

Фрагмент кода 21.1. Код Javascript `analyzeTemperature` для приложения-функции

Создает функцию. Каждое приложение-функция JavaScript начинается с экспорта функции, содержащей объект контекста. Этот объект используется для обмена данными.

Чтение содержимого сообщения из служебной шины

Создание объекта JSON декодированных сообщений служебной шины

Декодирование из формата base64

Извлечение зарегистрированной температуры из устройства Интернета вещей

Создание ответа для возврата в приложение логики

Вывод температуры в журнал консоли

```
module.exports = function (context, req) {
    var buffer = new Buffer(req.body.ContentData, 'base64')
    var decodedString = buffer.toString();
    var objects = JSON.parse(decodedString);
    var temperature = objects["temperature"];
    context.res = {
        body: {
            analysis: "Recorded temperature was " + temperature + "!"
        }
    };
    context.log("Recorded temperature was " + temperature);
    context.done();
};
```

Завершает функцию. Каждое приложение-функция JavaScript должно заканчиваться вызовом context.done, который сообщает приложению-функции о завершении кода.

13 Вернитесь в конструктор приложений логики для шага функции, затем выберите текстовое поле «Текст запроса» и «Сообщение служебной шины» в списке динамического контента справа.

14 В конструкторе приложений логики выполните перетаскивание для переупорядочения шагов так, чтобы действие «Отправить электронное письмо» оказалось ниже шага приложения-функции `analyzeTemperature` (см. рис. 21.13).

15 Выберите действие «Отправить электронное письмо», а затем — текстовое поле «Текст сообщения электронной почты».

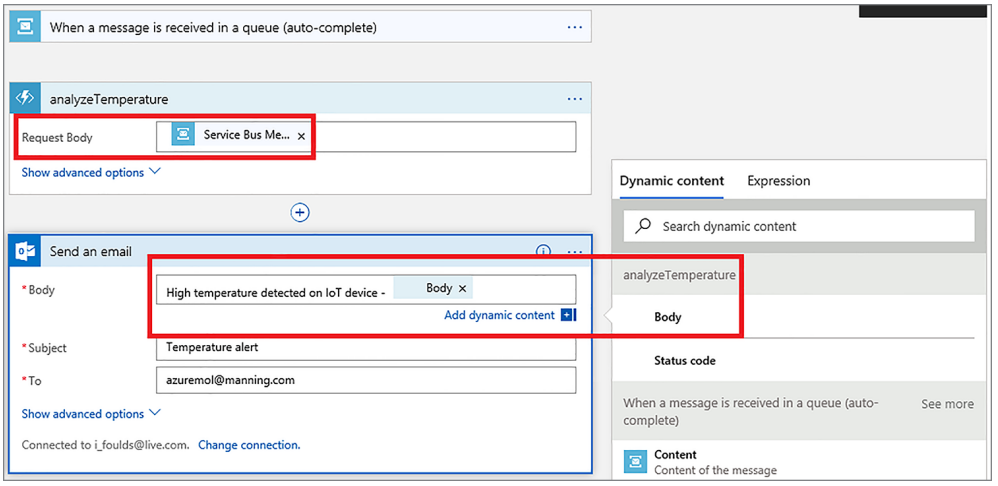


Рис. 21.13. Перетащите действие «Отправить электронное письмо» под функцию `analyzeTemperature`. Выберите конец текста сообщения. Откроется диалоговое окно динамического содержимого. Чтобы вставить значение температуры, вычисленное приложением-функцией, выберите текст сообщения в функции `analyzeTemperature`.

16 В функции `analyzeTemperature` выберите ответ «Текст» (см. рис. 21.13).

17 В конструкторе приложений логики нажмите «Сохранить».

У вашего бессерверного приложения много динамических компонентов. Прежде чем запускать моделируемое устройство Интернета вещей Raspberry Pi для создания оповещений по электронной почте с показаниями температуры, рассчитанными приложением-функцией, давайте посмотрим, что вы создали. На рис. 21.14 представлен обзор всех компонентов, которые теперь используются в бессерверном приложении.

Рис. 21.14. Все получаемые от моделируемого устройства Raspberry Pi сообщения, содержащие оповещение о температуре, направляются в конечную точку, которой является очередь сервисной шины. Сообщения в очереди служебной шины вызывают запуск приложения логики, которое передает сообщение приложению-функции. Функция JavaScript анализирует показание температуры и возвращает его приложению логики, которое затем отправляет уведомление по электронной почте с показанием температуры, зарегистрированным датчиком в устройстве Интернета вещей.

18 Откройте моделируемое устройство Raspberry Pi в веб-браузере и запустите приложение. При каждом создании оповещения о температуре приложение логики запускает приложение-функцию для извлечения данных о температуре из текста сообщения и включения их в уведомление по электронной почте. Чтобы показание температуры превысило 30 °C, после чего в сообщение вводится оповещение о температуре, может потребоваться несколько секунд. После ввода этого оповещения и обработки сообщения вы получите уведомление по электронной почте, в котором будет указана температура.

Сделайте глубокий вдох и похвалите себя. Вы сделали много!

Ошибки аутентификации из приложения логики в приложении-функции

Вы можете просмотреть журнал выполнения в окне обзора приложения логики на портале Azure. Если вы получаете много повторяющихся ошибок, выберите одну из них, чтобы узнать больше о том, где она происходит.

Распространенная проблема заключается в том, что приложение логики не авторизуется автоматически для взаимодействия с приложением-функцией. Повторное развертывание приложения логики обычно позволяет устранить эту ошибку, но чтобы действительно решить проблему, необходимо добавить так называемый ключ *функции* в заголовок приложения логики.

Чтобы получить его, выберите приложение-функцию, а затем выберите созданную функцию, например `analyzeTemperature`. Под параметром «Управление» можно просмотреть и скопировать ключ функции по умолчанию. Скопируйте его, вернитесь в приложение логики и откройте конструктор.

В функции `analyzeTemperature` выберите «Добавить параметр», а затем — «Добавить заголовок». Вам следует отправить определенный объем информации в начале вызова в приложение-функцию, которое отправляет ключ. Процесс противоположен вводу пары ключей, но сейчас введите `x-functions-key` в качестве ключа, а затем вставьте фактический ключ функции в качестве значения.

Обновление интеграции приложения логики и приложения-функции займет несколько минут. Затем в журнале запуска приложения логики должны появиться события, работающие правильно, и приложение должно начать отправлять уведомления по электронной почте.

21.5 Не прекращайте обучение

В этой главе содержится много новых концепций. Фактически, в последних нескольких главах представлено много новых идей и технологий! Не беспокойтесь, если вам пока трудно понять, как начать реализацию всех этих сервисов Azure — контейнеров, искусственного интеллекта и машинного обучения, а также бессерверных вычислений. Эти главы были предназначены для того, чтобы показать вам, что можно делать в Azure и что вы можете не только выполнить перенос устаревших приложений. Приступая к созданию и запуску приложений в Azure, воспользуйтесь возможностью модернизировать приложения и просматривать рабочие процессы управления и развертывания. Существует множество сервисов Azure, которые упрощают и ускоряют жизненный цикл приложений, поэтому вам не нужно зацикливаться на работе виртуальных машин из-за того, что они удобны для бизнеса.

Да, Azure предлагает множество новых прекрасных сервисов, но все они в значительной степени используют базовые компоненты инфраструктуры, которые обсуждались в части 1 этой книги. Разработчики могут начать применять новейшие подходы к созданию приложений, которые включают Kubernetes и бессерверные вычисления, а администраторы — продолжать использовать свои знания локальных центров обработки данных, а также основ облачных вычислений и методов устранения неполадок. По мере роста потребностей бизнеса Azure может поддерживать их.

В главе 1 я открыто и честно заявил, что не буду рассматривать все сервисы Azure. Существует множество других сервисов Azure, которые необходимо изучить и в результате лучше узнать о сервисах, рассмотренных в этой книге. Я надеюсь, что вы нашли по крайней мере несколько тем, которые вас интересуют и мотивируют к изучению. В число моих любимых тем входят масштабируемые наборы виртуальных машин, Cosmos DB и сервис Azure Kubernetes.

21.5.1 Дополнительные учебные материалы

Я предвзят, но думаю, что отличным местом для продолжения изучения Azure является https://docs.microsoft.com/azure. Здесь вы найдете всю основную документацию по сервисам Azure, руководства по архитектуре, справочные материалы и ресурсы SDK, а также примеры. Для каждого сервиса Azure доступен набор кратких справок, учебных пособий и примеров, а также предоставляются концептуальные сведения и практические инструкции.

Если вы настроены серьезно, существуют возможности сертификации по Azure. Доступны такие экзамены, как *Microsoft Azure Administrator (AZ-104), Microsoft Azure Architect Technologies and Design (AZ-303 and AZ-304)* и *Microsoft Azure Security Technologies (AZ-500)*. Эта книга и практические упражнения, которые вы выполнили, включали в себя многие области, которым посвящены эти экзамены. Однако вам потребуется изучить дополнительные области применения Azure AD и рекомендации по проектированию перед сдачей экзаменов. На сайте Microsoft Learn по адресу https://docs.microsoft.com/learn приведены дополнительные схемы обучения для различных вариантов сертификации по Azure, которые помогут вам подготовиться.

21.5.2 Ресурсы GitHub

В этой книге вы использовали примеры кода, шаблоны и примеры приложений из https://github.com/fouldsy/azure-mol-samples-2nd-edition. Эти примеры должны обновляться по мере выпуска новых версий интерфейса командной строки Azure. В репозитории GitHub также есть примеры и шаблоны PowerShell для всех упражнений. В этой книге основное внимание уделялось интерфейсу командной строки Azure в Azure Cloud Shell, но вы можете выполнить все упражнения с использованием PowerShell или шаблона.

Если вы заметили какие-либо проблемы с примерами, создайте заявку на портале GitHub по адресу https://github.com/fouldsy/azure-mol-samples-2nd-edition/issues. В Azure все меняется быстро, и я хочу быть уверен в постоянном наличии у вас последних рабочих примеров, которые помогут вам учиться. Не стесняйтесь также вносить предложения! Для всех документов Azure по адресу https://docs.microsoft.com/azure можно писать отзывы, сообщать о проблемах и предлагать изменения. Поэтому, изучая остальные предложения в Azure, вы можете принимать участие в доработках, а также помогать другим учиться и развиваться.

21.5.3 Заключение

Сделайте глубокий вдох и осознайте, что изменения теперь в порядке вещей. Новые функции и сервисы появляются практически ежедневно. Интерфейс платформы Azure, как и интерфейсы платформ всех других основных поставщиков облачных сервисов может выглядеть немного иначе, чем при его последнем использовании вами (час назад). Если у вас есть базовые фундаментальные навыки и знания, которые вы, надеюсь, получили из этой книги, вы можете адаптироваться ко всем новым возможностям, предлагаемым Azure, и развиваться вместе с ними. У вас всегда будет возможность изучить что-то новое, и мне хотелось бы услышать, что вы в конечном итоге создаете и запускаете в Azure!

указатель

Символы

Символ && 27
Объект $ResourceGroups 276
Объект $servicePrincipalConnection 275

A

аварийное восстановление (DR) 201
аварийные дампы 180
Автоматизация Azure 243, 269–283
  PowerShell DSC 278–282
    опрашивающие серверы сервиса автоматизации Azure и 280–282
    определение 280–282
  модули runbook 272–274
    выполнение 276–277
    пример 274–277
    просмотр выходных данных 276–277
  обзор 179, 269–274
  ресурсы 272–274
  создание учетных записей в 271–272
автоматическое переключение 46
агенты 180

Б

базовый план обслуживания 36
базы данных в Cosmos DB
  в Cosmos DB
    добавление глобальной избыточности 149–152
    заполнение 145–149
    создание 145, 149–152
  масштабирование 143–144
Бастион Azure 24, 115
безопасность 115
бесплатный/общий сервисный план 36
бессерверные вычисления 317–333
  платформы обмена сообщениями 319–325
  ресурсы GitHub 333
  создание приложений Logic Apps 325–328
  создание приложений-функций для анализа данных устройств Интернета вещей 328–331
блок сообщений сервера (SMB) 53
блокировки 79–80
боты для веб-приложений
  запуск с помощью LUIS 264–267
  создание с помощью LUIS 264–267
  создание 260
Брандмауэр Azure 238

B

вариант «Отключить диски» 199
веб-перехватчики 274
веб-сайты, запуск в Kubernetes 295–297
веб-сервер LAMP 27, 72
веб-серверы
  в действии 28–29
  установка 24–27
веб-трафик
  разрешение доступа к виртуальным машинам 27–29
  создание разрешающих правил 28
ветви, в Git 41
взаимосвязь с искусственным интеллектом 257–259
взвешенный метод маршрутизации, диспетчер трафика 163
виртуализация 10–11
виртуальное сетевое соединение (vNIC) 11
виртуальные машины для обработки и анализа данных (DSVM) 258
виртуальные машины серии B 18
виртуальные машины. *См.* ВМ
виртуальные сети 58–64
  внешние IP-адреса 62–64
  разрешение DNS 62–64
  сетевые адаптеры 61
  создание подсетей 59
  создание 59
виртуальные частные сети (VPN) 19, 36, 38

виртуальный жесткий диск (VHD) 53
виртуальный ЦП (vCPU) 11
ВМ (виртуальные машины)
  восстановление 198–201
    восстановление на уровне файлов 199
    полное восстановление виртуальной машины 199–201
  диагностика 175–177
  добавление дисков 50–52
  избыточность с помощью групп доступности 96–102
    домены обновления 97–98
    домены сбоя 96–97
  изменение размеров 126–127
  масштабирование вниз 127
  масштабирование по вертикали 125–127
  масштабируемые наборы 129–136
    создание правил автоматического масштабирования 133–136
    создание 131–133
    установка приложений 139
  назначение групп внутренним пулам 116–119
  настройка с использованием подсистем балансировки нагрузки 119–122
  настройка 15–20
    виртуальные сети 19–20
    образы виртуальных машин и 16–17
    размеры ВМ 17–18
    хранилище Azure 18–19
  освобождение 30
  открыть доступ для веб-трафика 27–29
    просмотр работы веб-сервера 28–29
    создание правил, разрешающих веб-трафик 28
  пары ключей SSH, создание для проверки подлинности 20–22
  подключение 120–122
    с помощью SSH 24–27
    с помощью агентов SSH 70–72
  получение секретов из виртуальных машин с удостоверениями MSI 224–229
  развертывание из шаблонов 102–105
  размеры 17
  распределение по группам доступности 98–101
  расширения диагностики 178
  создание 14–32, 69–70
    в браузерах 22
    в зонах доступности 95
    виртуальная машина Windows 29–30
    очистка ресурсов 30
    с подсистемой балансировки нагрузки 119–122
    сокращение затрат 18
    устранение неполадок Azure 31–32
  удаление 30
  установка веб-серверов 24–27
  хранилище 47–50
    временные диски 49–50
    выбор хранилища: стандартное или премиум 48–49
    диски с данными 49–50
    параметры кэширования дисков 50
    просмотр распределения по группам доступности 101–102
  шифрование 211–214
    лабораторная среда 214–215
    хранение ключей шифрования в Azure Key Vault 211–213
внедрение сертификатов 229–232
внешние IP-адреса 20, 62–64, 94, 108
внутренние пулы 107, 116–119
внутренний пул IP-адресов в подсистемах балансировки нагрузки 107
внутренняя подсистема балансировки нагрузки 108
восстановление виртуальных машин 198–201
  восстановление на уровне файлов 199
  полное восстановление виртуальной машины 199–201
восстановление на уровне файлов 199
временные диски 49–50
время жизни (TTL) 167
всемирное координированное время (UTC) 196
высокопроизводительные диски SSD 18

Г

географическая маршрутизация 163–164
геоизбыточное хранилище (GRS) 56
геоизбыточное хранилище с доступом для чтения (RA-GRS) 57
гибридная рабочая роль сервиса автоматизации 272
глобальная избыточность 149–152
глобальная маршрутизация, с диспетчером трафика 162–173
  глобальное распределение трафика до ближайшего экземпляра 167–173
  создание профилей диспетчера трафика 164–166
глобально распределенные данные 152–156
группировка ресурсов 80–81
группы безопасности сети. *См.* NSG
группы доступности 91
  избыточность для виртуальных машин с помощью 96–102
    домены обновления 97–98
    домены сбоя 96–97
  просмотр распределения виртуальных машин по 101–102
  распределение виртуальных машин по 98–101
группы ресурсов 315

Д

делегирование реальных доменов 160–162
диагностика виртуальных машин 175–177
диагностика загрузки 175–177
диапазоны IP-адресов 60
динамическое назначение 62
диски
  временные 49–50
  диски с данными 49–50
  добавление в виртуальные машины 50–52
  параметры кэширования 50
диски HDD категории «Стандарт» 18
диски SSD категории «Премиум» 18–19
диски с данными 49–50
Диспетчер трафика
  глобальная маршрутизация и разрешение 162–173

глобальное распределение трафика до ближайших экземпляров 167–173
область диагностики 183
развертывание веб-приложений 174
создание профилей 164–166
Диспетчер трафика Azure. *См.* Диспетчер трафика
добавочное резервное копирование 193
долгосрочная поддержка (LTS) 22
домены
обновление 97–98
реальные, делегирование в сервис DNS Azure 160–162
сбоя 96–97
домены обновления 96
домены сбоя 96–97

Ж

жизненные циклы приложений 76–77
журналы диагностики 42–44
журналы. *См.* журналы диагностики

З

зависимости 82
закрытый ключ пары ключей SSH 71
записи псевдонимов 160
записи сервера имен 160
записи сервиса 160
записи указателя 160
записи хоста IPv4 160
записи хоста IPv6 160
заполнение баз данных 145–149
запрос curl 226–227
запуск с помощью LUIS 264–267
захват сетевых пакетов 186–188
захват сетевых пакетов 186–188
защита
ресурсы 78–79
трафик с группами NSG 64–68
связывание NSG с подсетями 66–67
создание групп NSG (групп безопасности сети) 68–72
создание правил фильтрации NSG 67–68
защита ресурсов 79–80
защищенные виртуальные машины, удаление 205
защищенный трафик, создание веб-приложений 68–72
использование агентов SSH для подключения к виртуальным машинам 70–72
создание виртуальных машин 69–70
создание сетевых подключений удаленного доступа 68–69
знак вставки 27
зональные сервисы 93
зонды работоспособности
настройка 110–112
обзор 107
создание 110–112
зоны доступности 91
избыточность инфраструктуры 95
создание виртуальных машин 95
создание сетевых ресурсов 94–95

И

избыточность инфраструктуры с зонами доступности 95
создание виртуальных машин в зонах доступности 95
создание сетевых ресурсов в зонах доступности 94–95
избыточность
для виртуальных машин с помощью групп доступности 96–102
обзор 56–57
преимущества 90–91
избыточность. *См.* также избыточность инфраструктуры с зонами доступности
изменение размеров виртуальных машин 126–127
изолированные среды 36
ИИ (искусственный интеллект) 253–268
Azure Cognitive Services 259–260
LUIS 261–264
боты веб-приложений
запуск с помощью LUIS 264–267
создание с помощью LUIS 264–267
создание 260
машинное обучение и 254–259
обзор 254–255
инструменты для специалистов по обработке и анализу данных 257–259
инструменты сторонних производителей 86
интеграция служебной шины с центрами Интернета вещей 322–325
интерактивный доступ к консоли загрузки 176
интервал зондирования конечной точки 168
Интернет вещей Azure 300–316
создание приложений-функций для анализа данных устройств 328–331
интеграция со служебной шиной 322–325
обзор компонентов 315
обзор 300–302
потоковая передача данных центра в веб-приложения 309–315
Центр, централизованное управление устройствами 303–309
интерфейс командной строки (CLI) 12
инфраструктура как код (IaC) 82
Инфраструктура как сервис (IaaS) 9, 14, 33
искусственный интеллект 254, 256
искусственный интеллект. *См.* AI
исходный код 5

К

квоты 102, 132
кластеры с AKS 294–295 коллекции 146
ключи
создание хранилищ ключей 219–221
хранение ключей шифрования в Azure Key Vault 211–213
ключ функции 332

кнопка «Развернуть в Azure» 98
Кнопка «Управление доступом (IAM) 79
команда az cosmosdb show 152
команда az group create 131
команда az keyvault create 212
команда az keyvault secret show 221
команда az storage account create 210
команда az vm create 51, 95, 197
команда az vm disk attach 51
команда az vm list-sizes 127
команда az vm resize 127, 129
команда az vm show 105
команда az vm show 95
команда git push azure master 156
команда git push dev master 45
команда install 27
команда ssh-keygen 21
команды, перенос длинных строк 40
конечная точка событий 310, 312
конечные точки сервиса 146
конечные точки 323
контейнеры 146, 284–299
  ACI (экземпляр контейнера Azure) 289–292
  AKS (сервис Azure Kubernetes) 293–297
    запуск веб-сайтов в Kubernetes 295–297
    создание кластеров с 294–295
  обзор 284–288
концентраторы событий Azure 321–322
Концентраторы событий Azure 321–322
кэширование для чтения и записи 50

Л

локально избыточное хранилище (LRS) 56
локальный диспетчер конфигураций (LCM) 278

М

маркер SAS (подписанного URL-адреса) 87
маркер подписанного URL-адреса (SAS) 87
маршрутизация по производительности 163–164
маршрутизация прямого трафика с правилами преобразования сетевых адресов 114–116
масштабируемые приложения 124–140
  веб-приложения
    обзор 136–139
    по вертикали 127–128
  масштабирование
    базы данных 143–144
    виртуальных машин вниз 127
    виртуальных машин по вертикали 125–127
    изменение размеров виртуальных машин 126–127
    масштабирование вниз 127
    ресурсов по горизонтали 128–129
  масштабирование веб-приложений 136–139
  масштабируемые наборы, для виртуальных машин 129–136
    создание 131–133
    создание правил автоматического масштабирования 133–136
  преимущества 124–129
  масштабирования веб-приложений по вертикали 127–128
  масштабирования виртуальных машин по вертикали 125–127
  масштабирования ресурсов по горизонтали 128–129
масштабируемый набор с одной виртуальной машиной 130
машинное обучение. *См.* ML
метод маршрутизации по приоритету, диспетчер трафика 163
метрики производительности 178–182
метрики производительности 178–182, 188
модули 270
модуль nx 283
модули Runbook для сервиса автоматизации Azure 274–277
  выполнение 182
  выполнение 276–277
  обзор 272–274
  просмотр выходных данных 276–277
мониторинг 175
монолитное приложение 288

Н

Наблюдатель за сетью Azure 182–188
  захват сетевых пакетов 186–188
  проверка потоков IP-адресов 183–184
  просмотр действующих правил NSG 184–186
Наблюдатель сети 184
Настраиваемое расширение скриптов 179
настройка
  виртуальные машины с подсистемой балансировки нагрузки 119–122
  зонды работоспособности 110–112
Настройка требуемого состояния (DSC) 179, 278, 282–283
Настройка требуемого состояния PowerShell 278–282
  опрашивающие серверы сервиса автоматизации Azure 179, 280–282
  определение PowerShell. *См.* Azure PowerShell 280–282
начальные записи зоны (SOA) 160
неактивные данные 208
непрерывная интеграция (CI) 75
непрерывная поставка (CD) 75
неструктурированные данные 144

О

обзор 255–256
облака, защита информации 216–221
  создание хранилищ ключей и секретов 219–221
  хранилища ПО и модули HSM 217–218
обнаружение конечных точек 153
обновления 234–249
  JIT (just-in-time) 237–241, 249
  группы NSG центра безопасности Azure 234
  Управление обновлениями Azure 241–249
    OMS (Operations Management Suite) 243
    просмотр и применение обновлений 245–249

обновления JIT (just-in-time) 237–249
оболочка Bash 12
образы, ВМ 16–17
обратная косая черта 40, 68
окно «Обзор управления обновлениями» 243
Окно «Центр безопасности — обзор» 236
операционная система ЦОД (DC/OS) 293
оповещения о метриках 182
оповещения 178–182
оповещения 178–182
опрашивающие серверы 280–282
оркестратор контейнеров 293
открытый ключ пары ключей SSH 20–22, 71
ошибки проверки подлинности 332

П

память (vRAM) 11
параллельные виртуальные машины 102
Параметр -A 121
параметр enableHttpsTrafficOnly 210
параметр interval, зонды работоспособности 111
параметр --no-self-perms 220
параметр threshold, зонды работоспособности 111
параметр --zone 95
параметры 82, 84, 89
пары ключей SSH 20–22
пары ключей 20
переключение с предварительным просмотром 46
переменная access_token 226
переменная connectionString 307
переменная database_password 228
переменная iotconnectionstring 312
переменные 82, 84, 89, 271
планы App Service 35–38
платформа как сервис (PaaS) 10, 33
платформы обмена сообщениями 319–325
ПО как сервис (SaaS) 10
поддерживаемые языки 34–35
Подключение по протоколу удаленного рабочего стола (RDP) 20, 71
подключения 270
подсети
  связывание групп NSG 66–67
  создание 59
подсистема балансировки нагрузки для Интернета 108
политики 193–196
  RPO (целевая точка восстановления) 194–195
  RTO (целевое время восстановления) 195–196
политика кэширования «только для чтения» 50
пользовательские сертификаты SSL 207
портал Azure 12
последовательные виртуальные машины 102
потоки IP-адресов, проверка 183–184
потоковая передача данных центра Интернета вещей 309–315
потоковая передача файлов журнала 43
правила DenyAll 185
правила автоматического масштабирования 133–136
правило AllowAzureLoadBalancerInBound 67
правило AllowVnetInBound 67
правило default-allow-ssh 241
правило DenyAllInBound 67, 184
приемники 180
приложения
  Logic Apps 325–328
  жизненные циклы 76–77
  Приложения-функции 328–331
  сервисные планы 38
приложения для балансировки нагрузки 106–123
приложения. *См.* также веб-приложения Azure
Приложения-функции Azure 318
Приложения-функции 328–331
пример Google Карт 256
проверка потоков IP-адресов 183–184
проверка потоков IP-адресов 183–184
программные хранилища 217–218
проект Let's Encrypt 207
промежуточная среда 41
просмотр действующих правил NSG 184–186
просмотр обновлений 245–249
протокол мониторинга конечной точки 168
профили, в диспетчере трафика 164–166
прямой трафик, маршрутизация 114–116
пулы
  внутренние 116–119
  пулы внешних IP-адресов 108–110
пулы IP-адресов 107
пулы внешних IP-адресов 107–110

Р

рабочие области Log Analytics 243
рабочий слот 46
развертывание сайтов в формате HTML 39–42
разделение ролей 62
размеры виртуальных машин общего назначения 17
размеры виртуальных машин, оптимизированные для вычислений 17
размеры виртуальных машин, оптимизированные для хранения 17
размеры виртуальных машин, оптимизированные с учетом памяти 17
размеры ВМ по GPU 17
разрешение, с диспетчером трафика 162–173
  глобальное распределение трафика до ближайшего экземпляра 167–173
  создание профилей диспетчера трафика 164–166
разрешение DNS 62–64, 158
расписания резервного копирования 196–198
расписания 134, 270
расположения конечных точек 153
распределенные атаки типа «отказ в обслуживании» (DDoS) 182
расширения 305
редактор Visual Studio 85–86
реестр контейнеров Azure. *См.* ACR
резервные копии 191–204
  Azure Site Recovery 201–204
  сервис архивации Azure 191–201
    восстановление виртуальных машин 198–201
    политики и хранение 193–196
    расписания резервного копирования 196–198
режим на основе HTTP-пути, зонды работоспособности 110

режим на основе порта, зонды работоспособности 110
режим применения и автозамены, DSC 279
режим применения и мониторинга, DSC 279
режим сходства сеансов 112–113
режим только применения, DSC 279
ресурсы 5, 7
  защита с помощью блокировок 79–80
  защита 78–79
  масштабирование по горизонтали 128–129
  очистка 30
  с помощью тегов
    группировка 80–81
    управление 80–81
  управление 78–79
ресурсы в сервисе автоматизации Azure 272–274
речевой сервис 259
роль администратора доступа пользователей 78
роль владельца 78
роль участника веб-сайта 79
роль участника виртуальной машины 79
роль участника 78
роль читателя 78

C

сайты в формате HTML, развертывание 39–42
свойство «Текст сообщения» 56
секреты
  получение из виртуальных машин с удостоверениями MSI 224–229
  создание 219–221
Сети Azure 58–72
  защита и контроль трафика с помощью NSG 64–68
    связывание NSG с подсетями 66–67
    создание NSG 64–65
    создание правил фильтрации NSG 67–68
  компоненты виртуальной сети 58–64
    виртуальные сетевые адаптеры 61
    внешние IP-адреса 62–64
    разрешение DNS 62–64
    создание виртуальных сетей 59
    создание подсетей 59
  создание примеров веб-приложений с защищенным трафиком 68–72
    использование агентов SSH для подключения к виртуальным машинам 70–72
    создание виртуальных машин 69–70
    создание сетевых подключений удаленного доступа 68–69
платформа Azure
  инструменты управления 11–13
    Azure Cloud Shell 12–13
    портал Azure 12
    Azure PowerShell 13
    локальный интерфейс командной строки Azure CLI 13
  обзор 8–13
  виртуализация 10–11
  устранение неполадок 31–32
  хранилище 18
серверы баз данных, вертикальное масштабирование 126
Сервис Azure Kubernetes. *См.* AKS
сервис Content Moderator 259
сервис Personalizer 259
сервис автозаполнения Bing 259
Сервис архивации Azure 191–201, 243
  восстановление виртуальных машин 198–201
    восстановление на уровне файлов 199
    полное восстановление виртуальной машины 199–201
  политики и хранение 193–196
    RPO (целевая точка восстановления) 194–195
    RTO (целевое время восстановления) 195–196
  расписания резервного копирования 196–198
сервис компьютерного зрения 259
сервис компьютерного зрения 259
сервис машинного обучения Azure 258
Сервис метаданных экземпляров (IMDS) 222
сервис перевода текстов 259
сервис поиска 259
сервис пользовательского поиска Bing 259
сервис принятия решений 259
сервис распознавания говорящего 259
сервис распознавания лиц 259
сервисные планы для приложений 35–38
сервисный план Premium 36
сервисы, избыточные в пределах зоны 93
сертификаты 270
  внедрение 229–232
  создание 229–232
сетевой трафик
  маршрутизация 158–174
  управление 158–174
сетевые адаптеры (NIC) 61
сетевые адаптеры 61
сетевые пакеты 186–188
сетевые подключения удаленного доступа 68–69
сетевые ресурсы 94–95
Сетка событий Azure 320–321
  интеграция служебной шины с центрами Интернета вещей 322–325
  Концентраторы событий Azure 321–322
  обзор 317–319
  служебная шина Azure 321–322
  создание служебной шины 322–325
сети. *См.* сети Azure
Сетка событий Azure 320–321
Сетка событий Azure 320–321
сеть произвольной рассылки 160
синтаксический анализатор jq 226
система нумерации с отсчетом от нуля 99
система нумерации, с отсчетом от нуля 99
слоты развертывания 44–46
служебная шина
  интеграция с центрами Интернета вещей 322–325
  создание 322–325
Служебная шина Azure 310, 321–322
служебная шина Azure 321–322
служебные программы Azure DevOps 7
соглашения об уровне обслуживания (SLA) 247
Соединение по протоколу удаленного рабочего стола (RDP) 20, 71
создание с помощью LUIS 264–267
создание служебной шины 322–325

создание 260
создание ВМ в браузерах 22
  размеры виртуальных машин 17
  хранилище Azure 18
сообщения об ошибках 31
состояние deny 238
специалисты по анализу данных, инструменты 257–259
среды App Service 36
стандартные диски SSD 18–19
стандартные квоты 102
стандартный сервисный план 36
статическое назначение 63
структурированные базы данных SQL 142
структурированные данные 144
субъект-сервис 222

Т

типы записей Azure DNS 160
типы ресурсов 84–85
точки восстановления 193
трафик
  балансировки нагрузки 112–114
  глобальное распределение до ближайших экземпляров 167–173
  защита и контроль трафика с помощью групп NSG 64, 68
    связывание NSG с подсетями 66–67
    создание NSG 64–65
    создание правил фильтрации NSG 67–68
  маршрутизация прямого трафика с правилами преобразования сетевых адресов 114–116
  определение распределения с помощью правил подсистемы

У

удаление защищенных виртуальных машин 205
удаленная работа 41
узел бастиона 23–24
управление
  ресурсы 78–79
  трафик с группами NSG 64–68
    связывание NSG с подсетями 66–67
    создание NSG 64–65
    создание правил фильтрации NSG 67–68
управление доступом на основе ролей (RBAC) 78, 161, 211
Управление обновлениями Azure 241–249
  OMS (Operations Management Suite) 243
  просмотр и применение обновлений 245–249
управляемые диски 18
управляемые удостоверения, назначенные пользователем 222
управляемые удостоверения, назначенные системой 222
уровень виртуального сетевого адаптера 185
уровень группы безопасности приложения 185
уровень подсети 185
условия производительности, оповещения 181–182
установка веб-серверов 24–27
устранение неполадок 175
  диагностика виртуальных машин 175–177
  метрики производительности 178–182
  Наблюдатель за сетью Azure 182–188
    захват сетевых пакетов 186–188
    проверка потоков IP-адресов 183–184
    просмотр действующих правил NSG 184–186
  оповещения 178–182
  платформа Azure 31–32
учебные материалы 333
учетная запись Azure, создание 5–7
учетные записи
  в Cosmos DB
    создание 145–149, 152
    заполнение 145–149
    добавление глобальной избыточности в 149–152
  в сервисе автоматизации Azure, создание 271–272
учетные данные 270
учетные записи запуска от имени 272

Ф

Федеральный стандарт по обработке информации (FIPS) 218
фильтрация 67–68
форум, для этой книги 5
функция «Тестировать в веб-чате» 266
функция concat, диспетчер ресурсов 85
функция copy, диспетчер ресурсов 84
функция copyIndex() 84, 98, 102, 104

Х

хранение шаблонов 87
хранилища, ключ 219–221
хранилище
  доступность 56–57
  в Azure 18
  на виртуальных машинах 47–50
    временные диски 49–50
    выбор хранилища: стандартное или премиум 48–49
    диски с данными 49–50
    параметры кэширования дисков 50
  избыточность 56–57
  хранилище очередей 55–56
хранилище таблиц 52 теги
  группировка ресурсов 80–81
  управление ресурсами 80–81
хранилище (vDisk) 11
Хранилище Azure 47–57
  добавление дисков к виртуальным машинам 50–52
  преимущества 52–57
    доступности хранилища 56–57
    избыточности 56–57
    хранилища очередей 55–56
    хранилища таблиц 53–54
  хранилище виртуальных машин 47–50
    временные диски 49–50
    выбор хранилища: стандартное или премиум 48–49
    диски с данными 49–50

параметры кэширования дисков 50
хранение 193–196
RPO (целевая точка восстановления) 194–195
RTO (целевое время восстановления) 195–196
хранилище BLOB-объектов 52
Хранилище ключей Azure 216–233, 304
MSI (управляемые удостоверения сервисов) 221–229
внедрение сертификатов 229–232
защита информации в облаках 216–221
создание хранилищ ключей и секретов 219–221
хранилища ПО и модули HSM 217–218
обзор 211
создание сертификатов 229–232
хранение ключей шифрования в 211–213
хранилище очередей 53, 55–56
хранилище файлов 53

Ц

целевое время восстановления (RTO) 193
Центр безопасности Azure 234, 249

Ч

частные IP-адреса 108

Ш

шаблоны быстрого запуска Azure 7
шаблоны для Azure Resource Manager 81–87
создание нескольких однотипных ресурсов 84–85
создание 82–84
средства для создания 85–86
хранение 87
шифрование 206–215
SSE (Шифрование сервиса хранилища) 209–210
виртуальных машин 211–214
неактивных данных 208–209
обзор 206–208
хранение ключей в Azure Key Vault 211–213
Шлюз приложений Azure 108
Шлюз приложений 107–108

Э

экземпляр контейнера Azure. *См.* ACI
экземпляры, создание 290–292

Я

язык программирования Perl 34
язык программирования Python 28, 34
язык структурированных запросов (SQL) 53, 142
языковой сервис 259

A

ACI (экземпляр контейнера Azure) 284, 290, 292
ACR (реестр контейнеров Azure) 293
AKS (сервис Azure Kubernetes) 284, 293–297
запуск веб-сайтов в Kubernetes 295–297
просмотр информации 294
создание кластеров 294–295
Amazon Web Services (AWS) 191
API (интерфейсы прикладного программирования) 31
APT (Advanced Packing Tool) 27
AWS (Amazon Web Services) 191
Azure AD (Azure Active Directory) 222
Azure Application Insights 180
Azure CLI 7, 12–13, 28, 31, 81–82, 152, 161
Azure Cloud Shell 12–13
Azure Cognitive Services 259–260
Azure Cognitive Services 259–260
Azure DNS (сервис доменных имен) 158–162
Azure Front Door 163–164
Azure IoT Edge 304–305
Azure Logic Apps 318
Azure Monitor 243
Azure PowerShell 11, 13, 31, 81–82, 161
Azure Resource Manager 75–89
подход к 75–81
контролю и группировке ресурсов с помощью 80–81
тегов 80–81
проектированию на основе жизненного цикла приложений 76–77
защита ресурсов и управление ими 78–79
защита ресурсов с помощью блокировок 79–80
шаблоны для 81–87
создания 82–84
создания нескольких однотипных ресурсов 84–85
средства для создания 85–86
хранение 87
Azure Service Fabric 289
Azure Site Recovery 201–204, 243
Azure Web Apps 33–45
журналы диагностики, просмотр 42–44
запуск ботов с помощью LUIS 264–267
масштабирование 127–139
обзор 34–35
поддержка языков и сред 34–35
потоковая передача данных центра Интернета вещей Azure 309–315
развертывание приложения в веб-приложении с несколькими экземплярами 140
репликация между регионами Azure 203
слоты развертывания 35, 44–46
создание 37–42
развертывание образца сайта в формате HTML 39–42
создание простых веб-приложений 37
создание с защищенным трафиком 68–72
использование агентов SSH для подключения к виртуальным машинам 70–72
создание виртуальных машин 69–70
создание сетевых подключений удаленного доступа 68–69
создание ботов 260
управление 42–44

B

Building the Web of Things (Guinard and Trifa) 315

C

CD (непрерывная поставка) 75
CI (непрерывная интеграция) 75
CLI (интерфейс командной строки) 12
Cosmos DB 141–157
  доступ к глобально распределенным данным 152–156
  добавление глобальной избыточности 149–152
  обзор 141–144
    масштабирование баз данных 143–144
    структурированные базы данных (SQL) 142
    неструктурированные базы данных (NoSQL) 142–143
  развертывание веб-приложения 156–157
  создание и заполнение баз данных 145–149
  создание учетных записей и баз данных 145–152

D

DC/OS (операционная система ЦОД) 293
DDoS (распределенные атаки типа «отказ в обслуживании») 182
dependsOn 104
DKIM (DomainKeys Identified Mail) 160
Docker Swarm 293
Docker 284, 287
Dockerfile 291–292
DomainKeys Identified Mail (DKIM) 160
DR (аварийное восстановление) 201
DSVM (виртуальные машины для обработки и анализа данных) 258

E

ETW (трассировка событий для Windows) 180
ExpressRoute 19, 183

F

FQDN (полное доменное имя) 63

G

Gardner, Lyza Danger 315
Git 12
  обучение 37
  пароль, сброс 314
  развертывание образца сайта в формате HTML 39–42
GitHub
  обзор 39
  примеры Azure для быстрого начала работы 87
  репозиторий для этой книги 5
  ресурсы 333
  сервис автоматизации Azure и управление версиями 274
  учетная запись, создание 7
GPU (графический процессор) 267
GRS (геоизбыточное хранилище) 56
Guinard, Dominique D. 315

H

HashiCorp 86
HPC (высокопроизводительные вычисления) 267
HSM (аппаратные модули безопасности) 212, 217–218
HTTP 20, 168, 206
HTTPS 20, 168, 206
Hyper-V 15

I

IaaS (инфраструктура как сервис) 9, 14, 33–34
IaC (инфраструктура как код) 82
IIS (Internet Information Services) 29, 233
IMDS (сервис метаданных экземпляров) 222
Internet Information Services (IIS) 29, 233
IPv4-адреса 109
IPv6-адреса 109

J

JavaScript on Things (Gardner) 315
JSON (JavaScript Object Notation) 82–83, 86
JWT (JSON Web Token) 226

K

Kubernetes 293, 295–299
  См. также AKS
Kubernetes in Action (Luksa) 298

L

LCM (локальный диспетчер конфигураций) 278
Learn Docker in a Month of Lunches (Stoneman) 297
Learn Git in a Month of Lunches (Umali) 37
Linux
  запуск веб-приложений 34
  использование DSC 282–283
подсистемы балансировки нагрузки 94
  компоненты 106–119
    зонды работоспособности 110–112
    маршрутизация прямого трафика с правилами преобразования сетевых адресов 114–116
    назначение групп ВМ для внутренних пулов 116–119
    определение распределения трафика с помощью правил подсистемы балансировки нагрузки 112–114
    создание пулов внешних IP-адресов 108–110
  в действии 120–122

определение распределения трафика с помощью правил 112–114
создание и настройка виртуальных машин 119–122
Logic Apps 35, 182, 325–328
LRS (локально избыточное хранилище) 56
LTS (долгосрочная поддержка) 22
LUIS (Language Understanding Intelligent Service) 261–264
LUIS (Language Understanding Intelligent Service)
запуск ботов для веб-приложений 264–267
обзор 257–264
создание ботов для веб-приложений 264–267
Luksa, Marko 298

M

Marketplace, Azure 7
Maven 12
Message Analyzer, Microsoft 186
Microsoft Message Analyzer 186
ML (машинное обучение) 253–268
MOF-файл 280
MOF-файл 280
MSI (управляемые удостоверения сервисов) 221–229

N

NAT (преобразование сетевых адресов) 107, 114–116
NIC (сетевые адаптеры) 61, 117
NoSQL (неструктурированные базы данных) 142–143
NSG (группы безопасности сети) 20
в центре безопасности Azure 234
защита и контроль трафика 64–68
обзор 112
просмотр действующих правил 184–186
связывание с подсетями 66–67
создание правил фильтрации 67–68
создание 64–65, 118

O

OMS (Operations Management Suite) 243, 272

P

PaaS (платформа как сервис) 10, 33, 37, 137
PHP 34
PowerShell. *См.* Azure PowerShell

R

RA-GRS (геоизбыточное хранилище с доступом для чтения) 57
Raspberry Pi 306–309
RBAC (управление доступом на основе ролей) 78, 161, 184, 211
readLocations 153
Representational State Transfer (REST) 31
REST (Representational State Transfer) 31
REST API 161

S

SaaS (ПО как сервис) 10
Sender Protection Framework (SPF) 160
servicePrincipalName 224
SLA (соглашения об уровне обслуживания) 247
SMB (блок сообщений сервера) 53
SONiC (программное обеспечение для открытых сетей в облаке) 11
SPF (Sender Protection Framework) 160
SQL (язык структурированных запросов) 53, 142
SSE (шифрование сервиса хранилища) 209–210
SSL-сертификат 207
SSH (Secure Socket Shell)
агенты для подключения к ВМ 70–72
подключение к ВМ 24–27

T

Terraform 86
Trifa, Vlad M. 315
TTL (время жизни) 167

U

Ubuntu Linux 14, 26
Umali, Rick 37
UTC (всемирное координированное время) 196

V

VHD (виртуальный жесткий диск) 53
VMware 15
VPN (виртуальные частные сети) 19, 36, 38, 183

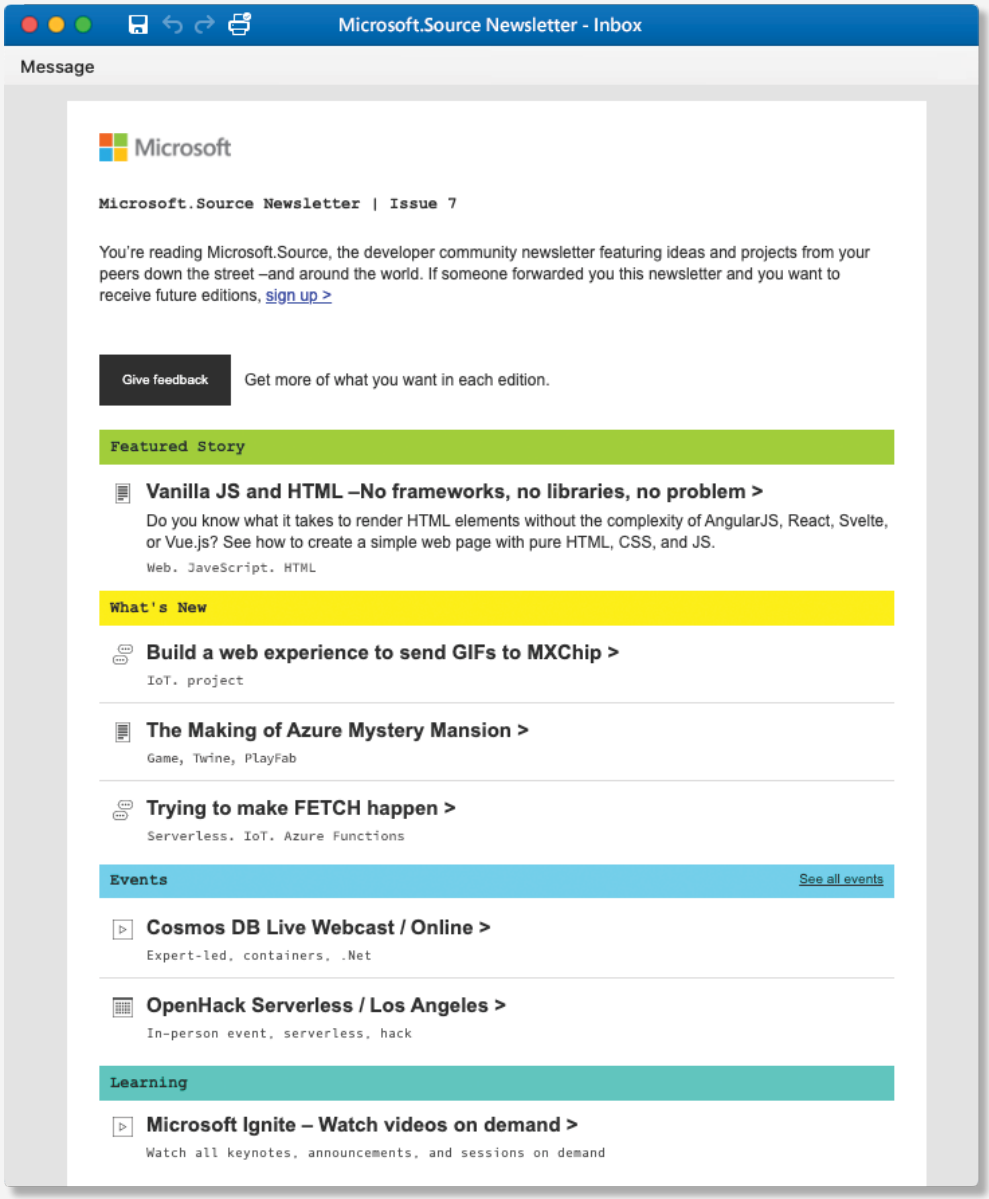
W

WebSocket 312
Windows, запуск веб-приложений 34

Z

ZRS (хранилище, избыточное в пределах зоны) 56

Microsoft



Создано разработчиками для разработчиков

Информационный бюллетень Microsoft.Source

Получите технические статьи, примеры кода и сведения о предстоящих мероприятиях с помощью Microsoft.Source — ежемесячного информационного бюллетеня для разработчиков.

- Следите за новейшими технологиями
- Общайтесь с коллегами на мероприятиях сообщества
- Учитесь с использованием с практических ресурсов

Зарегистрироваться