Интернет вещей (Internet of Things, IoT) – самая быстрорастущая технологическая отрасль. В промышленности технологии интернета вещей применяются для оптимизации оперативных расходов, увеличения срока эксплуатации продуктов и улучшения благосостояния людей.

Эта книга вам просто необходима, если вы хотите использовать весь спектр технологий для построения эффективной системы интернета вещей, включающей хоть одно устройство, хоть миллионы.

Мы начнем с изучения современных систем датчиков и сосредоточимся на их мощности и функциональности. После этого мы глубоко погрузимся в теорию коммуникаций, уделяя особое внимание персональным сетям ближнего диапазона, включая новые спецификации Bluetooth 5.0 и mesh-сети. Затем мы изучим связь по протоколу IP в локальных и глобальных вычислительных сетях, включая 802.11ah, 5G LTE, SigFox и LoRaWAN. Далее мы рассмотрим граничную маршрутизацию и шлюзы и их роль в туманных вычислениях, а также протоколы обмена сообщениями MQTT и CoAP.

Далее вы получите представление об облачных и туманных архитектурах, включая стандарты OpenFog. Мы завершим аналитическую часть книги применением статистического анализа, комплексной обработки событий и моделей глубокого обучения. И наконец проанализируем обзор стека безопасности и анатомию уязвимостей интернета вещей, а также противодействия угрозам программными ограничениями периметров и блокчейнов.

Вы узнаете:

- о роли и масштабе архитектуры для успешного развертывания системы интернета вещей, от датчиков до облака;
- о широком спектре технологий интернета вещей, охватывающих все: от датчиков до облачных вычислений;
- о компромиссах в выборе протоколов и коммуникаций в развертываемых системах интернета вещей;
- о необходимых навыках и специфических терминах, необходимых для работы в пространстве интернета вещей;
- об инженерных знаниях, необходимых для успешного построения систем интернета вещей.

ISBN 978-5-97060-672-8

Интернет-магазин: www.dmkpress.com
Книга – почтой: orders@alians-kniga.ru
Оптовая продажа: “Альянс-книга”
тел. (499)782-3889. books@alians-kniga.ru

Packt>

ДМК ПРЕСС ИЗДАТЕЛЬСТВО
www.дмк.рф

Архитектура интернета вещей

Архитектура интернета вещей

Перри Ли

ДМК ПРЕСС ИЗДАТЕЛЬСТВО

Перри Ли

Архитектура интернета вещей

Perry Lea

Internet of Things for Architects

Architecting IoT solutions by implementing sensors, communication infrastructure, edge computing, analytics, and security

BIRMINGHAM - MUMBAI

Перри Ли

Архитектура интернета вещей

Разработка архитектуры систем интернета вещей с применением датчиков, информационно-коммуникационной инфраструктуры, граничных вычислений, анализа и защиты данных

ДМК Пресс
издательство
Москва, 2019

УДК 004.738, 004.62
ББК 32.973
Л55

Ли П.
Л55 Архитектура интернета вещей / пер. с анг. М. А. Райтмана. – М.: ДМК Пресс, 2019. – 454 с.: ил.

ISBN 978-5-97060-672-8

Книга посвящена всем аспектам использования интернета вещей с примерами его применения в различных сферах, включая промышленность, умные города, транспортную систему и здравоохранение. Это наиболее подробное руководство в данной области на русском языке, которое рекомендуется всем отраслевым специалистам Ассоциацией интернета вещей в России. Вы получите полное представление об экосфере интернета вещей, различных технологиях и альтернативах, а также сможете рассмотреть IoT-архитектуру со всех ракурсов.

Издание будет полезно для архитекторов и проектировщиков информационных систем, технических специалистов и менеджеров по технологиям.

УДК 004.738, 004.62
ББК 32.973

ISBN 978-1-78847-059-9 (анг.)
ISBN 978-5-97060-672-8 (рус.)

Содержание

Дорогие друзья!

Книга Перри Ли «Архитектура интернета вещей» – это, пожалуй, на сегодня одно из самых подробных исследований темы интернета вещей с самых разных ракурсов и в самых разных аспектах: от влияния этого явления на общественные отношения до способов построения и программирования среды IoT.

Что самое приятное – книга к тому же еще и интересная, и любой фанат современных технологий, даже не слишком хорошо разбирающийся в инженерии или программировании, будет читать ее залпом. Но в первую очередь книга, разумеется, предназначена для технических специалистов, менеджеров технологий, программистов и проектировщиков, которые хотят лучше разобраться в сути интернета вещей, понять, как он устроен и какие имеет особенности. Очень здорово, что автор не просто рассказывает о IoT и его элементах, а дает живые примеры проектирования устройств и программирования, учит читателя практическим действиям, подсказывает, где получить дополнительные знания.

Это не просто книга, а учебник по интернету вещей для каждого. Ничего подобного в России до сих пор не выходило, и актуальность этой книги для нашей страны трудно преувеличить. Интернет вещей – основа цифровизации экономики, явление, которое в ближайшие годы до неузнаваемости может изменить промышленность, сельское хозяйство, энергетику да и всю нашу повседневную жизнь. Потребность в архитекторах IoT высока и благодаря книге Перри Ли специалисты в разных областях технологий получат знание о том, как все эти технологии могут образовать мир интернета вещей, и как построить этот мир вместе.

Уверен, что книга будет иметь огромный успех у нас в России, а Ассоциация интернета вещей, которую я представляю, будет рада рекомендовать книгу Перри Ли «Архитектура интернета вещей» своим членам, в том числе высшим учебным заведениям, где готовятся кадры будущей цифровой экономики страны и среди них – архитекторы интернета вещей.

Андрей Колесников,
директор Ассоциации интернета вещей

Об авторе

Перри Ли 21 год проработал главным ИТ-архитектором в компании Hewlett Packard, где проявил себя как замечательный технический специалист. Затем он перешел в Micron Technologies, где стал важным членом технического персонала и занял должность директора по стратегическому развитию, возглавив команду, занимающуюся продвинутыми вычислительными устройствами. В настоящий момент он является техническим директором в компании Cradlepoint, где в сферу его задач входит разработка и исследование в сфере интернета вещей и туманных вычислений.

Перри окончил Колумбийский университет по специальностям «информатика», «компьютерная инженерия» и «электроинженерия». Он старший член Института инженеров по электротехнике и электронике (IEEE) и главный спикер Ассоциации вычислительной техники (ACM). У него 8 патентов, и еще 40 патентов находятся на рассмотрении.

Благодарю свою жену Дон, а также родственников и друзей за поддержку, которую они оказывали мне в процессе написания этой книги.

Я хочу поблагодарить Сандру Капри из компании Ambient Sensors за критический обзор и комментарии по поводу датчиков и технологии ближней бесконтактной связи.

Я также благодарю Дэвида Раша из компании Cradlepoint за его комментарии относительно дальней бесконтактной связи и сотовых систем связи.

И, наконец, я благодарю разнообразные консорциумы и технические сообщества, такие как IEEE и ACM.

О рецензенте

Паркаш Карки – главный архитектор и руководитель отдела разработки с более чем 20-летним опытом работы в сфере информационных технологий. Он с отличием окончил Делийский университет, где изучал физику, и магистратуру по компьютерным приложениям, прошел сертификацию PMP (профессионал в управлении проектами) и получил ряд сертификатов Microsoft. Он преимущественно работал над различными технологиями Microsoft и технологиями с открытым исходным кодом, включая обширный опыт работы с DevOps и облачной платформой Azure. Как DevOps- и Azure-архитектор, он помогает клиентам перейти на эти технологии. Он с большим интересом относится к интернету вещей, искусственному интеллекту и средствам автоматизации.

Предисловие

Скорее всего, вы ежедневно сталкиваетесь с интернетом вещей в личной жизни и по работе. В основном, люди получают представление об интернете вещей благодаря личному опыту взаимодействия с фитнес-трекером Fitbit, умным динамиком Amazon Echo или термостатом от Google.

В 2017 г. при поиске по ключевому слову «IoT» сервис LinkedIn показывал 7189 объявлений о работе, тем или иным образом связанной с интернетом вещей. Портал Glassdoor показывает 5 440, а сайт **monster.com** – более тысячи объявлений. Рынок интернета вещей затягивает в себя все больше специалистов и технологий. Очень часто технические специалисты идут самым простым путем и подключают к интернету те объекты, которые раньше работали автономно. Безусловно, этот подход дает результаты, но это не то, что делает архитектор. Архитектор должен видеть общую картину, понимать, как работают разномастные технологии, разбираться в коэффициентах масштаба, безопасности и электроэнергии, – только тогда он сможет создать IoT-решение, которое будет не просто функционировать, но и приносить пользу компании, пользователям и акционерам.

Многие IoT-проекты оказываются провальными или застревают на стадии разработки по двум причинам. Во-первых, создать надежную систему сложно как в плане безопасности, так и в плане устойчивости к возникновению сбоев. Во-вторых, часто IoT-решение технически работает, но не получает одобрения со стороны менеджера, отвечающего за материально-техническое обеспечение ИТ-отдела. Поскольку границы интернета постоянно расширяются, мы, как архитекторы, должны учитывать тот факт, что миру корпоративных и промышленных информационных технологий уже более 50 лет. Конечно, IP-адрес можно присвоить даже лампочке, но будет ли от этого практическая польза для клиентов? В этой книге мы попытались подойти к интернету вещей с корпоративной/промышленной/коммерческой точки зрения, а не с развлекательной.

В этой книге интернет вещей рассматривается с точки зрения архитектуры и общего устройства, от датчика до облака, включая все физические переходы и трансформации между ними. Поскольку данная книга представляет собой пособие для архитекторов, мы попытались затронуть обозначенные вопросы достаточно глубоко, чтобы рассказать другим архитекторам что-то новое о трудностях и подводных камнях базовой системы. Об особенностях интернета вещей, например, о протоколе MQTT, разработке в облаке и методологии DevOps, проектировании источников питания и батарей, а также анализе радиосигналов, написано много книг и учебных пособий. Все это – важные компоненты IoT-системы, и квалифицированный архитектор, чтобы спроектировать надежную систему, должен принимать во внимание каждый из

этих элементов. Однако архитектор также должен чувствовать, когда пришла пора оторваться от деталей и сместить фокус своего внимания на архитектуру в целом.

От читателя не требуется хорошо разбираться в каждой области инженерии. В книге присутствует информация, касающаяся радиочастотных сигналов, питания и энергии и теории электрических цепей. Параллельно с этим книга повествует о программировании TCP/IP и управлении облаком. И, наконец, она подробно рассказывает о приложениях для машинного обучения, таких как сверточная нейронная сеть. Свести все эти технологии вместе – задача архитектора. Эта книга поможет вам развить свои навыки до необходимого уровня, но от вас не требуется глубоко разбираться в каждой из этих областей.

Возможности, которые предоставляет интернет вещей, невероятны: именно интернет вещей будет основой для следующего глобального переворота в промышленности, здравоохранении, системе государственного правления и предпринимательстве. Он, безо всякого сомнения, окажет сильное влияние на ВНП, наемный труд и рынки по всему миру. Однако, как вы увидите, с точки зрения безопасности с интернетом вещей связан ряд проблем и рисков.

Из тысяч объявлений о работе на тех сайтах многие предназначены для IoT-архитекторов, технических специалистов и координаторов, способных создавать IoT-решения, а не просто виджеты. Эта книга поможет вам изучить и освоить технологии, необходимые для выполнения такого рода проектов.

Кроме того, она интересная. Чтобы разработать устройство, с помощью которого можно регулировать освещение в доме или даже самолете или управлять тысячами уличных фонарей в городе на другом конце света, требуется невероятно мощная технология, созданная для фанатов технических новинок, но применяемая архитекторами.

Для кого эта книга?

Эта книга предназначена для архитекторов, проектировщиков систем, технических специалистов и менеджеров по технологиям, которые хотят получить представление об экосфере интернета вещей, различных технологиях и альтернативах, а также рассмотреть IoT-архитектуру со всех ракурсов.

О чем эта книга

Глава 1 «История интернета вещей» в описательной манере расскажет вам о развитии, значении и влиянии интернета вещей с исторической точки зрения. Также вы познакомитесь с примерами применения интернета вещей в различный сферах, включая промышленность, умные города, транспортную систему и здравоохранение.

Глава 2 «Архитектура и ключевые модули интернета вещей» создает общее представление о тех технологиях, о которых идет речь в данной книге, и о том,

как они сочетаются друг с другом. Каждый компонент служит своей цели и может подспудно оказывать влияние на другие компоненты. Эта глава очень важна для понимания общей картины взаимодействия связанных технологий. В этой главе также говорится о том, как можно повысить значимость интернета вещей.

Глава 3 «Датчики, оконечные точки и системы питания» рассказывает о миллиардах граничных IoT-устройств и сенсорных технологий, которые появятся в интернете. Говорится об основах разработки и проектирования датчиков и систем питания.

Глава 4 «Теория коммуникации и информации» представляет обзор важных сведений о динамической и математической моделях коммуникационных систем, которые имеют непосредственное отношение к интернету вещей. Вы познакомитесь с теоретическими основами, на которых строятся архитектурные решения в сфере телекоммуникаций.

Глава 5 «Беспроводная персональная сеть (WPAN) не на основе IP» рассказывает об основных технологиях и протоколах IoT, работающих не на основе IP. В этой главе подробно говорится о новой архитектуре Bluetooth 5, протоколах Zigbee, Z-Wave и сенсорных сетях с ячеистой топологией.

Глава 6 «WPAN и WLAN на базе IP» дополнит рассказ о системах ближней связи информацией о технологиях связи на основе протокола IP, включая стандарты 6LoWPAN, Thread и IEEE 802.11. В этой главе также подробно говорится о новых протоколах семейства 802.11, таких как протокол 802.11p для транспортных систем и протокол 802.11ah для интернета вещей.

Глава 7 «Системы и протоколы дальней связи (ГВС)» посвящена глобальной вычислительной сети и способам передачи данных от вещей в облако через системы дальней связи. В этой главе в подробностях рассказывается о стандарте сотовой связи LTE, технологиях LoRaWAN, Sigfox, а также о новом узкополосном варианте архитектуры сети LTE и о 5G-сетях.

Глава 8 «Маршрутизаторы и шлюзы» рассказывает о значимости периферийной маршрутизации и шлюзов. В этой главе рассматриваются системы маршрутизации, функции шлюза, технологии VPN, VLAN и приоритизация трафика, а также программно-конфигурируемые сети.

Глава 9 «IoT-протоколы передачи данных от граничного устройства в облако» знакомит вас с основными IoT-протоколами передачи данных в облако, такими как MQTT, MQTT-SN, CoAP, AMQP и STOMP. Вы узнаете, как их применять и, что важно, какие из них применять.

Глава 10 «Топология облачных и туманных вычислений» рассказывает об основах архитектуры облачных систем на примере OpenStack. Вы узнаете о том, какие тут существуют сложности, и о том, как туманные вычисления (на основе стандарта OpenFog) могут помочь в решении этих проблем.

Глава 11 «Анализ данных и машинное обучение в облачных и туманных платформах» описывает технологии и практические примеры эффективного анализа огромных массивов IoT-данных с помощью таких инструментов, как

обработка правил, обработка сложных событий и лямбда-выражения. В этой главе также изложена информация о приложениях машинного обучения для обработки IoT-данных и о том, в каких обстоятельствах следует их применять.

Глава 12 «Безопасность интернета вещей» рассматривает каждый описанный в данной книге компонент интернета вещей с точки зрения общей безопасности. Вы познакомитесь с теоретическими основами и архитектурой протоколов, аппаратными средствами, программно-определяемым периметром и с информацией об обеспечении безопасности технологии распределенных реестров.

Глава 13 «Консорциумы и сообщества» рассказывает о многочисленных промышленных, научных и государственных консорциумах, определяющих стандарты и правила в сфере интернета вещей.

Как получить от книги максимум пользы

В этой книге вы найдете несколько примеров проектирования устройств и программирования. Большинство примеров, связанных с программированием, представляют собой псевдокод на основе синтаксиса языка Python.Действующие примеры также основаны на Python 3.4.3, который подходит для macOS, Linux и Microsoft. По некоторым темам (например, глава 9) можно свободно найти библиотеки для Python (например, библиотека Paho, которая позволяет реализовать обмен данными по протоколу MQTT).

Тесное знакомство с некоторыми основополагающими методами математического анализа, теорией информации, электротехникой и информатикой поспособствует более глубокому пониманию интернета вещей с точки зрения архитектуры.

Ряд примеров из главы 10 берут за основу OpenStack или облачные сервисы Amazon AWS/Greengrass. Если вы хотите разобраться в задачах, стоящих перед архитектором в случаях из данных примеров, лучше завести аккаунт в облачном сервисе, хотя обязательным это не является.

Загрузите цветные изображения

Мы подготовили для вас PDF-файл с цветными изображениями из данной книги (снимки экрана и диаграммы). Если вы хотите скачать его, перейдите по ссылке: www.packtpub.com/sites/default/files/downloads/InternetofThingsforArchitects_ColorImages.pdf.

Условные обозначения

В данной книге используется ряд условных обозначений.

`Моноширинный текст`. Обозначает в тексте строки программного кода, названия таблиц с данными из баз. Пример: «Операция `insert` вносит изменение в оперативную память».

Фрагмент программного кода выглядит следующим образом:

```
rule "Furnace_On"
when
Smoke_Sensor(value > 0) && Heat_Sensor(value > 0)
then
insert(Furnace_On())
end
```

Когда мы хотим привлечь ваше внимание к тому или иному фрагменту программного кода, нужные строки или слова выделены полужирным шрифтом:

```
rule "Furnace_On"
when
Smoke_Sensor(value > 0) && Heat_Sensor(value > 0)
then
insert(Furnace_On())
end
```

Текст, который вводится или отображается в командной строке, выглядит следующим образом:

```
aws greengrass create-function-definition --name "sensorDefinition"
```

Полужирный шрифт. Обозначает новый термин, важное слово или слова, которые вы видите на экране, а также реальные и выдуманные URL-адреса, введенный пользователем текст и имена пользователей в Twitter. Например, текст меню или диалоговых окон будет выделен полужирным шрифтом. Вот пример: «**Internet Key Exchange (IKE)** – это протокол безопасности в IPsec».

Курсивом обозначены имена папок и файлов, расширения файлов, а также путь к файлам.

Предупреждения или важные примечания отмечены подобным образом.

Советы и подсказки выделены как этот текст.

Отзывы и пожелания

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

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

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Скачивание исходного кода примеров

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте www.dmkpress.com на странице с описанием соответствующей книги.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг — возможно, ошибку в тексте или в коде — мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от огорчения и поможете нам улучшить последующие версии этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу dmkpress@gmail.com, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в Интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Packt очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в Интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты dmkpress@gmail.com со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Глава 1

История интернета вещей

В четверг, 17 мая 2022 г., вы, как обычно, просыпаетесь около 6:30 утра по тихоокеанскому времени. Вы никогда не заводите будильник, вы один из тех людей, у кого хорошо развиты «биологические часы». Мгновение спустя перед вашими глазами предстает фантастическое солнечное утро, поскольку температура за окном достигает 70 °C. Ваш день будет абсолютно не похож на утро среды 17 мая 2017 г. Абсолютно все: образ жизни, здоровье, финансы, работа, транспорт, даже парковочное место, – будет другим. Все в мире вокруг вас изменится: энергия, медицина, сельское хозяйство, промышленность, транспортная система, общественный транспорт, экология, система безопасности, магазины и даже одежда. Причиной тому будет подключение повседневных объектов к интернету, или интернет вещей (IoT). Наиболее удачным (с моей точки зрения) названием этого явления стало бы «интернет всего».

Еще до того, как вы проснулись, в окружающем вас интернете вещей произошло множество событий. Датчик сна или умная подушка зафиксировали то, как вы спите. Данные были отправлены шлюзу IoT, а затем переданы вашему бесплатному облачному сервису, который отправляет отчеты на информационную панель на вашем телефоне. Вы обходитесь без будильника, но, если у вас, например, самолет в 5 утра, вы все же поставите будильник, который, опять же, контролируется облачным сервисом, работающим на протоколе **IFTTT** (англ. «**if this then that**» – «если это, тогда то»). Ваш двухзонный котел отопления обслуживается другим облачным сервисом и подключен к вашей домашней 802.11 Wi-Fi-сети, как и датчики дыма, дверной звонок, система полива, дверь гаража, камеры наблюдения и система безопасности. Ваша собака чипирована датчиком для отслеживания, снабженным источником электропитания, с помощью которого открывается дверка для домашних животных, а также позволяющим вам в любой момент узнать, где собака находится.

У вас больше нет компьютера как такового. Конечно, у вас есть планшетник и смартфон, поскольку это базовые устройства, но *центром* вашего мира стали очки VR/AR Goggles, поскольку экран намного лучше и больше. У вас в шкафу есть шлюз для туманных вычислений. Он подключен к провайдеру интернета 5G и к глобальной вычислительной сети, потому что проводное соединение не соответствует вашему образу жизни – вы мобильны, на связи и онлайн, неза-

висимо от того, где находитесь, а 5G-интернет и любимый оператор обеспечивают вам одинаковый уровень комфорта и в отеле в Майами, и в вашем доме в Бойсе, штат Айдахо. Кроме того, шлюз выполняет за вас множество бытовых задач, таких как обработка видео с камер наблюдения, чтобы определить, не произошел ли в доме как-нибудь сбой или инцидент. Система безопасности проверяет дом на предмет аномалий (странных звуков, возможных протечек воды, перегоревших ламп или испорченной вашей собакой мебели). Граничный узел также служит домашней док-станцией, где хранится резервная копия информации с вашего телефона, поскольку вы частенько разбиваете свои телефоны, а также выступает в качестве частного облака, даже если вы ничего не знаете об облачных сервисах.

До работы вы добираетесь на велосипеде. Ваша веломайка оснащена печатными датчиками и отслеживает частоту вашего сердцебиения и температуру тела. Эти данные с помощью технологии Bluetooth с низким энергопотреблением (Bluetooth Low Energy) передаются на ваш смартфон, параллельно с этим вы слушаете музыку, причем Bluetooth-гарнитура получает аудиосигнал также с помощью технологии Bluetooth. По пути вы проезжаете мимо несколько рекламных щитов, транслирующих видео и рекламные ролики в режиме реального времени. Вы заезжаете в местную кофейню, у входа в которую установлено цифровое информационное табло, которое, обращаясь к вам по имени, спрашивает, хотите ли вы повторить свой вчерашний заказ: *большая чашка американо со сливками*. Это возможно благодаря радиомаяку и шлюзу, которые с расстояния полутора метров позволяют определить, что вы приближаетесь к табло. Разумеется, вы выбираете вариант «*да*». Большинство людей добирается до работы на машинах, а оптимальное парковочное место им позволяют подобрать умные датчики, которым оснащены все парковки. Конечно же, вы оставляете свой велосипед в наиболее подходящем месте – прямо у въезда на парковку, на велосипедной стоянке.

Компания, в которой вы работаете, участвует в природосберегающей программе потребления возобновляемой энергии. Корпоративная политика придерживается курса на сведение к нулю вредных выбросов и отходов, являющихся результатом эксплуатации офисных помещений. В каждой комнате установлены датчики присутствия, дающие информацию не только о том, что в комнате кто-то есть, но и о том, кто именно. Ваш именной бейдж, служащий пропуском на работу, представляет собой устройство-маяк с батарейкой, заряда которой хватит на 10 лет. О вашем присутствии становится известно, как только вы подходите к входной двери. Освещение, вентиляция, система отопления и система кондиционирования воздуха, автоматические шторы, потолочные вентиляторы и даже цифровые информационные табло связаны между собой. Центральный узел туманных вычислений получает всю информацию о здании и синхронизирует ее с облачным сервером. За принятие решений в режиме реального времени отвечает процессор правил, который принимает в расчет количество людей в помещении, время суток, время года, а также температуру

внутри помещения и снаружи. Для максимально эффективного использования энергетических ресурсов система опирается на условия окружающей среды. На главных предохранителях также установлены датчики, которые *отслеживают* тип энергопотребления, передавая эту информацию на узлы туманных вычислений, чтобы при появлении странных тенденций система обратила на них внимание.

Все это происходит благодаря нескольким алгоритмам машинного обучения и граничной аналитики в режиме реального времени, которые отрабатываются в облаке и выполняются на граничных устройствах. В офисном помещении также расположена малая сота 5G для подключения к оператору верхнего уровня, а кроме того, несколько шлюзов для малых сот, обеспечивающих связь внутри здания. Внутренние шлюзы для малых сот 5G также выступают в роли локальной вычислительной сети.

Ваш телефон и планшет поймали внутренний сигнал 5G, и вы подключились к оверлейной сети, определяемой вашим программным обеспечением, и тут же оказались в корпоративной локальной вычислительной сети. Ваш смартфон выполняет за вас большой объем задач, по сути, он стал вашим персональным шлюзом для подключения к вашей собственной персональной сети, окружающей ваше тело. Вы спешите на свою первую сегодняшнюю встречу, но ваш коллега еще не пришел и появляется на пару минут позже вас. Он извиняется, но объясняет, что добирался на работу не без приключений. Его новая машина отправила производителю сообщение о вероятных отклонениях в работе компрессора и турбонагнетателя. Производитель немедленно отреагировал и позвонил владельцу, чтобы поставить того в известность о 70%-ной вероятности поломки турбокомпрессора в течение ближайших двух дней эксплуатации машины. Они договорились о времени встречи в дилерском центре, куда были отправлены новые детали для ремонта компрессора. Это сэкономило вашему коллеге приличную сумму, которую пришлось бы потратить на замену турбокомпрессора, а также позволило избежать массы побочных последствий.

Обедать ваша компания решила в центре города, в новом ресторанчике с рыбными тако. Вы вчетвером втискиваетесь в купе, где и вдвоем-то не очень просторно, и отправляетесь в путь. К сожалению, вам приходится припарковаться на одной из самых дорогих многоуровневых парковок. На этой парковке действует динамическое ценообразование, т. е. цена зависит от спроса и количества свободных мест. В результате некоторых событий и повальной тупости людей стоимость парковки увеличилась вдвое, несмотря на то, что сейчас полдень вторника. Хороший момент в том, что та же система, которая отвечает за ценообразование, передает вашей машине и смартфону информацию о свободных парковках и их точное местонахождение. Вы вводите адрес рыбного тако-ресторана, высвечивается информация о подходящей парковке и ее загруженности, и вы бронируете место еще до своего приезда туда. Машина подъезжает к воротам, которые идентифицируют вас по сигналу смартфона и открываются. Вы заезжаете на парковку, и мобильное приложение отмечает

в облаке парковки, что вы поставили машину на правильное место, над нужным датчиком.

После обеда вам нужно посетить производственный объект на другом конце города. Это типичная фабрика: несколько литьевых машин, подъемно-транспортные устройства, упаковочные машины и вся сопутствующая инфраструктура. Недавно качество продукции стало снижаться. У конечного продукта появились проблемы с узловым соединением, и он стал менее привлекательным внешне по сравнению с прошлым месяцем. Вы приехали на точку, побеседовали с менеджером и проинспектировали фабрику. Все кажется нормальным, но качество, безусловно, изменилось в худшую сторону. Вы вдвоем садитесь и начинаете разбираться с панелями управления производственного этажа.

Для контроля над цехом система опирается на ряд датчиков (датчик вибрации, температуры, скорости, обзора и слежения). Данные поступают и визуализируются в режиме реального времени. Несколько алгоритмов диагностического обслуживания проверяют различные устройства на внешние признаки износа или сбоев. Эта информация направляется производителю оборудования, а также вашей команде. Анализ журналов учета и анализ динамики показателей не выявил никаких аномалий, его выполняли ваши лучшие специалисты. Все указывает на то, что на решение этой проблемы уйдет не одна неделя и лучшим сотрудникам вашей организации придется ежедневно проводить рабочие совещания.

Однако в вашем распоряжении большой объем данных. Все данные по производственному цеху за долгое время хранятся в базе данных. Это дорогая услуга, расходы на которую поначалу было трудно объяснить, но вы считаете, что в данных обстоятельства она оправдает себя на тысячу процентов. Прогнав все ретроспективные данные через сложную систему анализа данных и событий, вы быстро получаете набор критериев, влияющих на качество проблемных деталей. Отследив ситуацию до тех событий, которые привели к потере качества, вы понимаете, что причиной стал не точечный сбой, а целый ряд факторов:

- внутренняя температура рабочего пространства была повышена на 2 °C в целях энергосбережения в летний период;
- в результате экономии электроэнергии скорость работы сборочного конвейера снизилась на 1,5%;
- одна из литьевых машин должна была в ближайшее время проходить плановое профилактическое техобслуживание, а температура рабочего пространства и скорость конвейера привели к тому, что она дала сбой раньше, чем прогнозировалось.

Вы выявили проблему и перенастроили алгоритмы диагностического обслуживания с учетом изменившихся параметров, чтобы в будущем избегать подобных ситуаций. В целом, рабочий день был продуктивным.

Неважно, сбудется или не сбудется эта выдуманная история – как бы там ни было, она довольно близка к современной реальности. Википедия (ru.wikipedia.org/wiki/интернет_вещей) определяет понятие «интернет вещей» следующим

образом: «Интернет вещей (IoT) (англ. Internet of Things, IoT) – концепция вычислительной сети физических предметов («вещей»), оснащенных встроенными технологиями для взаимодействия друг с другом или с внешней средой, рассматривающая организацию таких сетей как явление, способное перестроить экономические и общественные процессы, исключающее из части действий и операций необходимость участия человека».

История развития интернета вещей

Термин «интернет вещей», по всей видимости, обязан своим появлением Кевину Эштону, который в 1997 г., работая на компанию Proctor and Gamble, для управления системой поставок применил технологию радиочастотной идентификации (RFID). Благодаря этой работе в 1999 г. его пригласили в Массачусетский технологический институт, где он с группой единомышленников организовал исследовательский консорциум Auto-ID Center (более подробную информацию можно найти на сайте www.smithsonianmag.com/innovation/kevin-ashton-describes-the-internet-of-things-180953749/). С тех пор интернет вещей совершил переход от простых радиочастотных меток к экосистеме и индустрии, которая к 2020 г. привлечет, создаст или поглотит 5 трлн долларов из 100 трлн мирового ВВП, т. е. 6% мирового ВВП. Вплоть до 2012 г. идея подключения вещей к интернету преимущественно относилась к смартфонам, планшетам, ПК и ноутбукам. По сути, к тем вещам, которые во всех отношениях выступают в качестве компьютера. До этого, с момента появления первых робких зачатков интернета (таких как созданная в 1969 г. сеть ARPANET), большинства технологий, на которых строится интернет вещей, просто не существовало. До 2000 г. большинство устройств, которые можно было подключить к интернету, представляло собой компьютеры различных размеров. Таблица 1.1 демонстрирует постепенное подключение *вещей* к интернету.

Таблица 1.1. История интернета вещей

Год	Устройство	Источник
1973	Марио У. Кардулло получает патент на первую радиочастотную метку	США, патент US 3713148 A
1982	Подключенный к интернету автомат с газированной водой в университете Карнеги-Меллон	www.cs.cmu.edu/~coke/history_long.txt
1989	Подключенный к интернету тостер на конференции Interop '89	Журнал IEEE Consumer Electronics Magazine (Том: 6, выпуск: 1, январь 2017)
1991	Компания HP представила HP LaserJet IIISi: первый подключенный к сети Ethernet сетевой принтер	hpmuseum.net/display_item.php?hw=350
1993	Подключенная к интернету кофеварка в Кембриджском университете (первая подключенная к интернету камера)	www.cl.cam.ac.uk/coffee/qsf/coffee.html
1996	Подразделение General Motors OnStar (дистанционная диагностика 2001)	en.wikipedia.org/wiki/OnStar

Таблица 1.1 (окончание)

Год	Устройство	Источник
1998	Появление организации Bluetooth SIG	www.bluetooth.com/aboutus/our-history
1999	Холодильник LG Internet Digital DIOS	www.telecompaper.com/news/lg-unveils-internetready-refrigerator--221266
2000	Первые проявления разработанной компанией HP концепции всепроникающей компьютеризации (Cooltown): HP Labs, система вычислительных и коммуникационных технологий, которые в сочетании друг с другом создают подключение к интернету для людей, мест и объектов	www.youtube.com/watch?v=U2AkkuIVV-I
2001	Выпуск первого устройства, использующего технологию Bluetooth: мобильный телефон KDDI с поддержкой Bluetooth	edition.cnn.com/2001/BUSINESS/asia/04/17/tokyo.kddibluetooth/index.html
2005	Международный союз электросвязи, специализированное учреждение ООН, выпустил отчет, в котором впервые были сформулированы прогнозы развития интернета вещей	www.itu.int/osg/spu/publications/internetofthings/InternetofThings_summary.pdf
2008	Появление первого IoT-сообщества IPSO Alliance, целью которого было содействие подключению вещей к интернету	www.ipso-alliance.org
2010	Успешная разработка полупроводниковых светодиодных ламп привела к развитию концепции умного освещения	www.bu.edu/smartlighting/files/2010/01/BobK.pdf
2014	Компания Apple создала протокол iBeacon для маячков	support.apple.com/ru-ru/HT202880

Безусловно, понятие «интернет вещей» вызывает большой интерес и пристальное внимание. Это легко заметить, хотя бы исходя из того, что, начиная с 2010 г., количество получаемых патентов бурно растет (www.uspto.gov). Количество поисковых запросов в системе Google (trends.google.com/trends/) и публикаций в коллегиально рецензируемом журнале IEEE резко поползло вверх в 2013 г. (см. рис. 1.1).

Перспективы развития интернета вещей

Интернет вещей захватит практически каждый сегмент в сфере промышленности, бизнеса, здравоохранения и потребительских товаров. Важно понимать последствия, а также то, почему эти совершенно различные отрасли будут вынуждены изменить свой подход к производству товаров и предоставлению услуг. Вероятно, вы как архитектор будете иметь дело с каким-то одним конкретным сегментом, однако вам не помешает понимание того, как различные сферы экономики могут взаимно влиять друг на друга в остальных случаях.

Как говорилось ранее, согласно распространенному мнению, интернет вещей и связанные с ним сферы услуг, отрасли промышленности и торговли к 2020 г. (предположительно) затронут своим влиянием от трех (The route to a trillion devices, ARM Ltd 2017: community.arm.com/cfs-file/__key/telligent-

evolution-components-attachments/01-1996-00-00-00-01-30-09/ARM-_2D00_-The-route-to-a-trillion-devices-_2D00_-June-2017.pdf) до четырех процентов (The Internet of Things: Mapping Value Beyond the Hype, McKinsey and Company 2015: www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/The%20Internet%20of%20Things%20The%20value%20of%20digitizing%20the%20physical%20world/Unlocking_the_potential_of_the_Internet_of_Things_Executive_summary.ashx) мирового ВВП. Мировой ВВП за 2016 г. составил 75,64 трлн долларов США, а, по некоторым оценкам, к 2020 г. он вырастет до 81,5 трлн долларов. Таким образом, в интернет вещей будет вовлечено от 2,4 до примерно 4,9 трлн долларов.

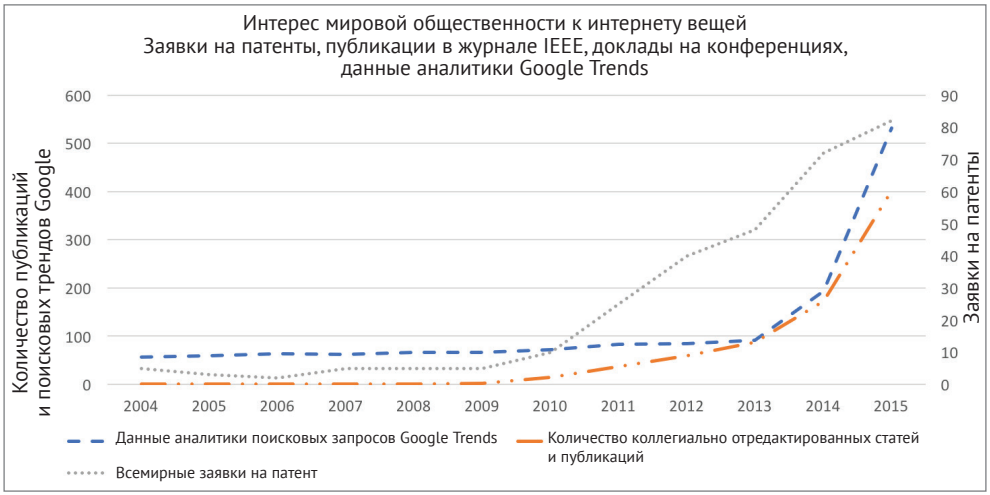


Рис. 1.1 ❖ Анализ ключевых слов при поиске информации об интернете вещей, патентах и технических публикациях

Численность взаимосвязанных объектов беспрецедентна. Размышляя о развитии этой сферы, невозможно не задуматься о сопряженных рисках. Чтобы попробовать сгладить возможные последствия, возьмем несколько исследовательских компаний и их отчетов о том, сколько объектов будет подключено к 2020 г. Разброс очень большой, но все же порядок величин примерно одинаков. В среднем, согласно этим 10 аналитическим прогнозам, к 2020–2021 гг. будет 33,4 млрд подключенных к интернету объектов. Недавно корпорация ARM провела исследование и предсказала, что к 2035 г. подключенным к интернету будет 1 трлн устройств. Судя по всему, соответствующие проекты в ближайшем будущем будут развиваться и наращивать свой потенциал со скоростью 20% в год (рис. 1.2).

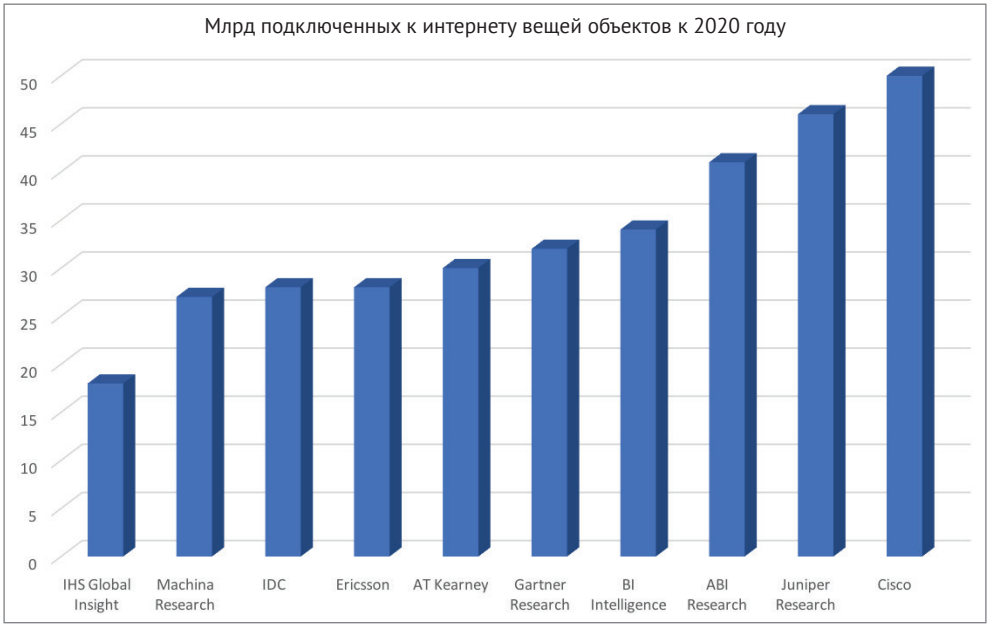


Рис. 1.2 ❖ Количество подключенных к интернету объектов по оценкам различных аналитиков и корпораций

Эти цифры должны впечатлить читателя. Например, если за основу взять очень консервативную точку зрения, в соответствии с которой к интернету будет подключено только 20 млрд устройств (исключая традиционную вычислительную технику и мобильные устройства), получится, к интернету каждую секунду будут подключаться 211 новых объектов.

Для электронной промышленности и сферы информационных технологий эти данные имеют огромное значение, поскольку ежегодный прирост населения Земли на данный момент составляет приблизительно 0,9–1,09% (esa.un.org/unpd/wpp/). Темп роста населения Земли достиг своего пика в 1962 г., когда он составлял 2,6% в год, и с тех пор под влиянием ряда факторов медленно снижается. Первый и основной фактор – улучшение экономических показателей и повышение мирового ВВП отрицательно сказались на рождаемости. К другим факторам относятся войны и голод. Эта тенденция подразумевает, что количество объектов, связанных с людьми, перестанет расти, а основной объем подключенных к интернету устройств будут составлять подключенные к интернету объекты и объекты с межмашинной коммуникацией. Это важно, поскольку в сфере информационных технологий главным фактором ценности сети является не количество размещенных в ней данных, а количество подключений. Именно так гласит закон Меткалфа, о котором мы поговорим далее в данной книге. Также следует отметить, что после того, как в 1990 г. организа-

ция ЦЕРН (CERN) запустила первый интернет-сайт, количество пользователей сети Интернет выросло до 1 млрд человек всего за 15 лет. Интернет вещей, по оценкам, будет расти со скоростью 6 млрд подключенных устройств в год. Это, конечно, станет огромным фактором влияния (рис. 1.3).

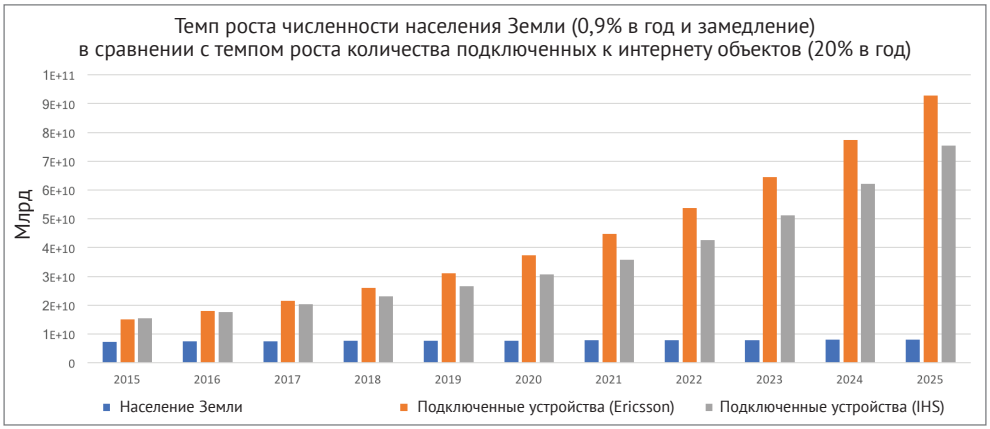


Рис. 1.3 ❖ Дисбаланс между ростом численности населения Земли и ростом количества подключенных к интернету вещей. Наблюдается следующая тенденция: ежегодный прирост подключенных к интернету объектов составляет 20% против 0,9% ежегодного прироста населения. Люди больше не будут основным показателем пропускной способности сети и успешности ИТ-проекта

Необходимо отметить, что с экономической точки зрения изменится не только способ получения дохода. Влияние интернета вещей или любой другой технологии проявляется в виде:

- новых источников дохода (получение электроэнергии экологически чистым методом);
- сокращения расходов (уход за пациентами на дому);
- сокращения срока вывода продукта на рынок (автоматизация производства);
- усовершенствования структуры цепочки поставок (учет материальных активов);
- сокращения производственных потерь (кража, порча товаров с коротким сроком годности);
- повышения производительности (машинное обучение и анализ данных);
- вытеснения (умный термостат Nest вытесняет с рынка обычные термостаты).

Читая эту книгу, в первую очередь необходимо помнить о той дополнительной ценности, которую привносят IoT-решения. Если это просто новый гаджет, объем рынка будет ограничен. Направление будет развиваться и приносить хо-

рошие плоды, только если ожидаемые преимущества перевешивают возможные издержки. В целом, целевая технология должна быть на пять порядков лучше обычной технологии. Именно к этому я и стремился, работая в сфере информационных технологий. Прикидывая затраты, необходимые для внесения изменений, обучение, распространение, техническую поддержку и пр., необходимо исходить из принципа 5-кратного улучшения.

Теперь подробно поговорим об отдельных отраслях промышленности и того, как на них повлияет интернет вещей.

Индустрия и производство

Промышленный интернет вещей (**Industrial IoT**, **IIoT**) – это один из наиболее крупных и быстро развивающихся сегментов интернета вещей с точки зрения количества подключенных устройств и степени полезности этих сервисов для производства и автоматизации предприятий. Этот сегмент традиционно служит операционно-технологической базой. Сюда входят аппаратные и программные средства мониторинга физических устройств. Традиционные задачи информационных технологий решаются иначе, чем операционно-технологические задачи. Операционные технологии (OT) сосредоточены на оценке производительности, времени безотказной работы, сборе данных и ответной реакции в режиме реального времени, а также безопасности систем. Информационные технологии направлены на безопасность, группирование, сервисы и предоставление данных. Поскольку интернет вещей начинает занимать важное место в сфере производства и промышленности, миры ИТ и OT объединятся, особенно в области диагностического обслуживания тысяч производственных машин и станков, и смогут обеспечивать беспрецедентным объемом данных частные и публичные облачные инфраструктуры.

К характеристикам этого сегмента относится необходимость предоставлять операционно-технологической системе готовые решения в режиме реального времени или почти в режиме реального времени. Это означает, что во всем, что касается производственного цеха, главным параметром для интернета вещей будет время отклика. Кроме того, важнейшую роль будут играть длительность простоя и безопасность. Это подразумевает потребность в запасе мощности и, вероятно, в наличии частных облачных сетей и хранилищ данных. Промышленный интернет вещей – это один из наиболее быстро развивающихся сегментов на этом рынке. Важной особенностью этого направления является то, что оно опирается на *старые* технологии, т. е. на аппаратные и программные средства, которые нельзя назвать актуальными. Часто 30-летние производственные станки работают на серийных интерфейсах RS485, а не на современной беспроводной ячеистой архитектуре.

Примеры и результаты применения промышленного интернета вещей

Примеры и результаты применения промышленного интернета вещей включают в себя следующее:

- профилактическое обслуживание нового и использовавшегося ранее промышленного оборудования;
- рост производительности благодаря спросу в реальном времени;
- энергосбережение;
- системы безопасности, такие как измерение температуры, замер давления и контроль над утечкой газа;
- экспертная система для производственного цеха.

Потребитель

Потребительские устройства были одной из первых категорий предметов, подключаемых к интернету. Потребительский интернет вещей начался с подключенной к интернету кофеварки в одном университете в 1990-х гг. Он расцвел с распространением технологии Bluetooth в начале 2000-х гг. Теперь миллионы домов оснащены термостатами Nest, лампочками Hue, виртуальным голосовым помощником Alexa и ТВ-приставками Roku. Кроме того, люди пользуются браслетами Fitbit и другими портативными устройствами. Потребительский рынок обычно первым перенимает все новые технологии. Также мы можем рассматривать эти устройства как гаджеты. Все они поставляются в аккуратной упаковке и обертке, и, в основном, все они действуют по принципу «вставь и включи».

Одна из сложностей потребительского сегмента заключается в бифуркации стандартов. Например, мы видим, что в основе некоторых протоколов беспроводной персональной сети лежат стандарты Bluetooth, Zigbee и Z-wave (которые не являются интероперабельными).

У этого направления также очень много общего с медицинским сегментом, куда относятся специализированные портативные устройства и домашние системы наблюдения за состоянием здоровья. Пока мы оставим их в стороне, отметим только, что медицинский сегмент будет развиваться и не станет ограничиваться простыми домашними приборами медицинской диагностики (например, функционалом браслетов Fitbit).

Примеры применения пользовательского интернета вещей

Вот несколько примеров применения пользовательского интернета вещей:

- **умные устройства для дома**: система полива, гаражные двери, замки, фонари, термостаты и система охраны;
- **портативные устройства**: трекеры здоровья и движения, умная одежда/аксессуары;
- **животные**: системы отслеживания местонахождения домашних животных, умные двери для собак.

Розничная торговля, финансы и маркетинг

Эта категория относится к любой области, где осуществляется розничная торговля. Это может быть магазин или торговая палатка. Кроме того, эта катего-

рия также тесно связана с финансовыми организациями и сферой маркетинга. Сюда входят традиционные банковские и страховые услуги, а также досуговый и гостиничный бизнес. Интернет вещей в области розничной торговли уже оказывает влияние на эту сферу, его задача – снизить издержки реализации и повысить качество обслуживания. Для реализации этой задачи существует множество IoT-инструментов. Чтобы избежать излишней сложности, рекламу и маркетинг мы также отнесем к данной категории.

В этом сегменте ценность выражается в немедленных финансовых операциях. Если интернет вещей не приносит этого результата, необходимо пересмотреть целесообразность вложений в IoT-решения. Это привносит дополнительные сложности в виде необходимости находить новые способы снижения издержек или повышения доходов. Если покупатели смогут эффективнее решать свои задачи, продавцы товаров и услуг смогут обслуживать их быстрее и обходиться меньшим штатом сотрудников.

Примеры применения интернета вещей для розничной торговли

Ниже приведены некоторые примеры применения нтернета вещей для розничной торговли:

- целевая реклама, например поиск фактических или потенциальных покупателей в непосредственной близости и предоставление им информации о товаре/услуге;
- оповещения, например маркетинговый анализ на основе таких данных, как распознавание приближения клиента, схема движения и интервалы времени;
- учет материальных активов, в частности инвентаризация, управление ущербом и оптимизация системы поставок;
- контроль над холодильниками, в частности оценка состояния хранящейся в холодильниках скоропортящейся продукции. Применение прогнозной аналитики к продовольственным товарам;
- страхование материальных активов;
- оценка страховых рисков водителей;
- цифровые вывески в торговых точках, гостиницах и по городу;
- системы оповещения в развлекательных заведениях, на конференциях, концертах, парках развлечений и музеях.

Медицина

Сфера медицины может соперничать с промышленностью и логистикой за лидирующие позиции в том, что касается доходности и влияния IoT-решений. Практически в каждой развитой стране любая система, которая повышает качество жизни и позволяет сократить расходы, является высшим приоритетом. Интернет вещей в ближайшем будущем сможет обеспечить дистанционное и многофункциональное наблюдение за пациентами, где бы те ни находились. Высокотехнологичные средства аналитики и машинного обучения будут спо-

собны диагностировать заболевания и назначать лекарства. Такие системы также будут сигнализировать о ситуациях, когда человеку жизненно необходима помощь. В настоящий момент в мире существует около 500 млн портативных медицинских датчиков, а в ближайшем будущем это число увеличится более чем на 10%.

С медицинским сегментом связаны существенные трудности. Согласно закону HIPAA, регламентирующему доступ к медицинской информации, IoT-системы должны квалифицироваться как больничные инструменты и оборудование. Портативные и домашние системы должны обмениваться данными с медицинскими учреждениями 24/7, должны быть надежными и работать без задержек и сбоев. Системы могут работать и на стороне больницы, если требуется отслеживать состояние пациента в процессе его транспортировки в спецмашине.

Примеры применения интернета вещей в медицине

Вот некоторые примеры применения интернета вещей в медицине:

- уход за пациентом на дому;
- модели обучения в предиктивной и превентивной медицине;
- наблюдение и уход за пожилыми пациентами и пациентами с деменцией;
- учет больничного оборудования и ресурсов;
- контроль и обеспечение безопасности лекарственных препаратов;
- дистанционная медицинская помощь;
- исследование медикаментов;
- индикаторы падения пациента.

Транспортировка и логистика

Транспортная сфера и логистика станут важной, если не основной, областью применения интернета вещей. К примерам применения интернета вещей в данной сфере относится отслеживание доставляемого, перемещаемого или транспортируемого груза, какой бы транспорт ни использовался: фура, поезд, самолет или корабль. Сюда же относится подключение к интернету транспортных средств, благодаря чему они могут предлагать водителю помощь или осуществлять профилактический ремонт и обслуживание вместо водителя. В данный момент любое среднестатистическое транспортное средство, купленное новым, оснащено приблизительно 100 датчиками. Эта цифра удвоится, когда такие функции, как связь между машинами, связь между машиной и дорожной инфраструктурой и автоматическое вождение, станут обязательным условием безопасности и комфорта. Все это распространяется не только на частные транспортные средства, но и на железнодорожный транспорт и морские грузоперевозки, где нет возможности для простоя. Еще один вариант применения – техническая помощь на дорогах, возможность отслеживать ситуацию со служебными автомобилями. Некоторые случаи применения мо-

гут быть очень простыми, но при этом очень дорогостоящими, например, отслеживание местоположения свободных автомобилей аварийно-ремонтной службы. Необходимы также системы автоматического построения маршрута для служебных автомобилей и технического персонала в зависимости от ситуации на дороге.

Эта мобильная категория отличается тем, что здесь важную роль играет геолокация. Основная часть геолокационных данных поступает через GPS-навигацию. Интернет вещей опирается на такие данные, как ресурсы и время, а в этом случае и пространственные координаты.

Примеры применения интернета вещей в сфере транспортировки и логистики

Вот несколько примеров применения интернета вещей в сфере транспортировки и логистики:

- отслеживание перемещений и местонахождения автомобилей из парка;
- идентификация и отслеживание железнодорожных вагонов;
- учет грузов и комплектования транспортных единиц в парке;
- планово-предупредительный ремонт автомобилей на дороге.

Сельское хозяйство и окружающая среда

Интернет вещей в области сельского хозяйства и окружающей среды включает в себя такие направления, как ветеринария, анализ земли и почвы, анализ микроклимата, эффективное водопотребление и даже предсказания возможных катастроф (геологических или погодных катаклизмов). Несмотря на то, что темпы роста численности населения Земли снижаются, страны становятся все более экономически устойчивыми. Голод и недостаток питания встречается все реже. Иначе говоря, спрос на производство продуктов питания удвоится к 2035 г. Интернет вещей может способствовать мощному прорыву в сельском хозяйстве. Благодаря умному освещению, подстраивающемуся под возрастные потребности птицы, можно повысить показатели прироста и понизить смертность поголовья на птицефабрике. Кроме того, умные системы освещения могут ежегодно экономить 1 млрд долларов на электроэнергии (по сравнению с обычным нерегулируемым освещением лампами накаливания). Другой вариант применения IoT-систем заключается в отслеживании состояния здоровья поголовья фермы, опираясь на данные о перемещении и расположении датчиков. Животноводческое хозяйство сможет выявлять животных с потенциальными заболеваниями, прежде чем бактериальная или вирусная инфекция успеет распространиться. Системы анализа данных на граничных устройствах позволят находить, выявлять и изолировать животных в режиме реального времени, опираясь на аналитику или машинное обучение.

Особенностью этого сегмента является его удаленность (например, вулканы) или низкая плотность населения (кукурузные поля). Это оказывает влияние на системы обмена данными, о которых мы поговорим позже, в главах 5 и 7.

Примеры применения интернета вещей в сельском хозяйстве и окружающей среде

Вот несколько примеров применения интернета вещей в сельском хозяйстве и окружающей среде:

- умные системы полива и удобрения для повышения урожайности;
- умное освещение в птицеводческих хозяйствах и фермах для повышения поголовья;
- ветеринария и мониторинг состояния здоровья скота;
- планово-предупредительный ремонт оборудования удаленных ферм под контролем производителя;
- съемка земель с помощью беспилотных летательных аппаратов;
- оптимизация цепочки поставок фермерской продукции на рынок с учетом материальных активов;
- автоматизация ферм;
- мониторинг вулканической активности и геологических разломов для прогнозирования катаклизмов.

Энергетика

Энергетический сегмент включает в себя мониторинг вырабатываемой источником электроэнергии и ее потребление. Большое количество исследований и разработок посвящено потребительским и коммерческим устройствам мониторинга электроэнергии, таким как умные электросчетчики, которые задействуют маломощные протоколы широкого спектра действия и измеряют потребление электроэнергии в реальном времени.

Множество электростанций расположено в удаленных или неблагоприятных регионах, например, в пустынных местностях (солнечная энергия), холмистых областях (ветроэлектростанции) или представляют собой опасные территории (ядерные реакторы). При этом данные должны обрабатываться в режиме реального времени или практически реального времени, чтобы можно было моментально отреагировать на срочные сигналы систем управления (общая черта с промышленным сегментом). Это может повлиять на принципы применения интернета вещей в данной сфере. Необходимость обработки данных в реальном времени мы обсудим далее в этой книге.

Примеры применения интернета вещей в энергетике

Вот несколько примеров применения интернета вещей в энергетике:

- анализ данных с нефтедобывающих платформ (тысячи датчиков и точек измерения) с целью повышения производительности;
- удаленный мониторинг и обслуживание солнечных панелей;
- оценка аварийной опасности атомных электростанций;
- общегородские умные электросчетчики для оценки уровня энергопотребления и спроса на электроэнергии;
- регулирование угла атаки лопатки турбины на ветряных энергетических установках в реальном времени в зависимости от погоды.

Умный город

Умный город – это определение, описывающее объединение в общую систему того, что раньше существовало автономно. Умные города – это один из наиболее быстро развивающихся сегментов, в котором соотношение доходов и расходов очень показательно, особенно если мы посмотрим на налоговые поступления. Умные города непосредственно затрагивают качество жизни горожан, в частности, в отношении безопасности, защищенности и простоты получения услуг. Например, некоторые города, такие как Барселона, полностью оборудованы этой системой, и мусорные баки там опустошаются в зависимости от текущей наполненности, а также от времени последнего вывоза мусора. Это позволяет более эффективно справляться с вывозом мусора, задействуя меньше ресурсов, расходуя меньше налогов и при этом своевременно предотвращая появление неприятных запахов гниения органических отходов. Умные города подчиняются правительственным постановлениям и предписаниям (как мы увидим далее), поэтому этот сегмент имеет точки пересечения с правительственной сферой.

Одной из отличительных черт умного города можно считать большое количество датчиков. Например, чтобы на каждом пересечении улиц в Нью-Йорке установить умную камеру, потребуется более 3000 камер. Или, например, в Барселоне и подобных городах необходимо около миллиона экологических датчиков для отслеживания объема потребления электроэнергии, данных о температуре воздуха, условий окружающей среды, качества воздуха, уровня шума и информации о загруженности парковочных мест. Все они передают сигнал по более узкой полосе радиочастот по сравнению с камерами видеонаблюдения, но общий объем передаваемых данных будет практически таким же, как у уличных камер в Нью-Йорке. Эти особенности – большой объем данных и узкополосная связь – необходимо учитывать при разработке оптимальной IoT-архитектуры.

Примеры применения интернета вещей для системы «умный город»

Вот несколько примеров интернета вещей для системы «умный город»:

- контроль над уровнем загрязнения и анализ регулирующего воздействия путем обследования состояния окружающей среды;
- микроклиматические прогнозы погоды с опорой на городскую сеть датчиков;
- повышение эффективности и снижение расходов за счет вывоза и переработки мусора по необходимости, а не по графику;
- улучшение ситуации на дорогах и экономия топлива за счет умных светофоров и разметки;
- рациональное потребление электроэнергии благодаря городскому освещению по необходимости;
- оптимизация снегоуборочных работ благодаря поступающим в реальном времени данным о ситуации на дорогах, погодных условиях и ближайших снегоуборочных машинах;

- умная система полива в парках и общественных местах, учитывающая погодные условия и текущее состояние;
- умные камеры наблюдения для отслеживания преступных деяний и автоматизированная система оповещений AMBER Alert в реальном времени;
- умные парковки, помогающие автоматически подобрать лучшее парковочное место;
- мониторинг износа и состояния мостов, улиц и городской инфраструктуры, направленный на своевременное обслуживание и продление срока службы.

Правительство и армия

Муниципальное правительство, правительство штатов и федеральное правительство, а также армия очень заинтересованы в тех технологиях, которые предлагает интернет вещей. Взять хотя бы принятый в Калифорнии подзаконный акт B-30-15 (www.gov.ca.gov/news.php?id=18938), согласно которому к 2030 г. выбросы парниковых газов должны быть снижены на 40% по сравнению с 1990 г. Для достижения таких сложных целей необходимы мониторы окружающей среды, системы датчиков энергии и машинный интеллект, с помощью которых можно будет по мере необходимости менять схемы потребления электроэнергии, но и не израсходовав на это весь бюджет Калифорнии. Другой пример – проекты типа «Интернет военных вещей» (Internet of Battlefield Things), цель которого – улучшить условия для военнослужащих и повысить эффективность противостояния врагам. Этот сегмент тесно связан с концепцией умного города, поскольку мониторинг государственной инфраструктуры осуществляется так же, как мониторинг трасс и мостов.

Роль правительства в интернете вещей также очень существенна, она проявляется в виде стандартизации, распределении частотного спектра и правилах. Например, посмотрите, как распределяется и защищается диапазон частот, предоставляемый разным провайдерам. В этой книге мы поговорим о том, как под контролем федерального правительства развивались некоторые технологии.

Примеры применения интернета вещей в правительстве и армии

Вот несколько примеров применения интернета вещей в правительстве и армии:

- оценка вероятности террористической угрозы посредством маяков и IoT-устройств, осуществляющих системный анализ;
- дистанционно управляемые разведывательные аппараты и датчики;
- применение сенсорных бомб в условиях боевых действий, формирование системы приборов обнаружения для выявления угроз;
- системы учета государственных материальных активов;
- системы отслеживания и обнаружения военнослужащих в реальном времени;

- комплексные датчики для мониторинга ситуации в опасной обстановке;
- мониторинг уровня воды для оценки состояния плотин и предотвращения затоплений.

Заключение

Добро пожаловать в мир интернета вещей. Мы как архитекторы в этой новой сфере должны понимать, чего хочет клиент, и какие технологии необходимы в данном случае. IoT-системы нельзя просто запустить и забыть. У запрыгивающего на поезд интернета вещей клиента всегда есть ряд ожиданий.

Во-первых, должно появиться дополнительное преимущество. Каким оно будет, зависит от вашей работы и от того, какие задачи стоят перед клиентом. По моему опыту, целью всегда является 5-кратное улучшение показателей, что часто наблюдается при внедрении новых технологий в существующие отрасли. Во-вторых, IoT-проект, по своей природе, подразумевает подключение множества устройств. Преимущество интернета вещей заключается не в том, что какое-то одно устройство или какая-то одна точка передает данные серверу. Данные передает целая группа устройств, и ценность представляет совокупная информация со всех этих устройств. В любом проекте должен быть заложен потенциал к развитию, это необходимо учитывать при разработке чего-то нового.

Теперь перейдем к топологии IoT-системы в целом, а затем рассмотрим ее отдельные компоненты. Не забывайте, данные – это нефть нового времени.

Глава 2

Архитектура и ключевые модули интернета вещей

Экосфера интернета вещей начинается с простейших датчиков, расположенных в самых удаленных уголках света, и преобразует аналоговое физическое воздействие в цифровые сигналы (язык интернета). Затем данные совершают сложное путешествие по проводным и беспроводным сигналам, различным протоколам, через естественные помехи и наложения электромагнитных полей, в результате которого попадают в интернет. Оттуда пакеты данных по различным каналам передаются в облако или в крупный центр обработки данных. Сильная сторона интернета вещей заключается в том, что он представляет собой не просто единичный сигнал от одного датчика, а сумму всех сигналов с сотен, тысяч, возможно, миллионов датчиков, точек и устройств.

Эта глава начинается со сравнительного описания архитектуры интернета вещей и архитектуры сетей межмашинного взаимодействия. В ней также говорится о том, какая роль принадлежит архитектору в процессе построения способной к росту,безопасной и крупномасштабной архитектуры интернета вещей. Для этого архитектор должен быть способен четко сформулировать, какую пользу проект принесет клиенту. Архитектор также должен уметь решать инженерные и вспомогательные задачи, маневрируя между возможными вариантами реализации проекта.

В данной книге освещается широкий спектр вопросов, таких как трансформация физических величин в цифровые сигналы, системы энергоснабжения и аккумулирование энергии, управление миллиардом устройств на расстоянии метра, километра, системы и протоколы передачи информации на максимальную дальность, теория сетей и информации, интернет-протоколы для IoT, роль граничной маршрутизации и шлюзов. Далее в книге говорится о работе с данными в процессе облачных и туманных вычислений, а также о продвинутом машинном обучении и комплексной обработке событий. В завершение повествования мы поговорим о безопасности и уязвимости крупнейшей мишени для атак на Земле.

Экосистема интернета вещей

Эти сферы активно пользуются тем множеством устройств, программного обеспечения и сервисов, которое предлагает интернет вещей. Практически каждая крупная технологическая компания вкладывает или вкладывала деньги в интернет вещей. Уже успели сформироваться новые рынки и технологии (а некоторые из них успели потерпеть фиаско или были перепроданы). В этой книге мы поговорим практически о каждом сегменте информационных технологий, поскольку у каждого из них своя роль в интернете вещей:

- **датчики**: встроенные системы, операционные системы реального времени, источники бесперебойного питания, микроэлектромеханические системы (МЭМС);
- **системы связи между датчиками**: зона охвата беспроводных персональных сетей составляет от 0 см до 100 м. Для обмена данными между датчиками применяются низкоскоростные маломощные информационные каналы, которые часто построены не на протоколе IP;
- **локальные вычислительные сети**: обычно это системы обмена данными на основе протокола IP, например, 802.11 Wi-Fi-сеть для быстрой радиосвязи, часто это пиринговые или звездообразные сети;
- **агрегаторы, маршрутизаторы, шлюзы**: поставщики встроенных систем, самые бюджетные составляющие (процессоры, динамическая оперативная память и система хранения данных), производители модулей, производители пассивных компонентов, производители тонких клиентов, производители сотовых и беспроводных радиосистем, поставщики межплатформного программного обеспечения, разработчики инфраструктуры туманных вычислений, инструментарий для граничной аналитики, безопасность граничных устройств, системы управления сертификатами;
- **глобальная вычислительная сеть**: операторы сотовой связи, операторы спутниковой связи, операторы маломощных глобальных сетей (Low-Power Wide-Area Network, LPWAN). Обычно применяются транспортные протоколы интернета для IoT и сетевых устройств (MQTT, CoAP и даже HTTP);
- **облако**: инфраструктура в качестве поставщика услуг, платформа в качестве поставщика услуг, разработчики баз данных, поставщики услуг потоковой и пакетной обработки данных, инструменты для анализа данных, программное обеспечение в качестве поставщика услуг, поставщики озер данных, операторы программно-определяемых сетей/программно-определяемых периметров, сервисы машинного обучения;
- **анализ данных**: огромные массивы информации передаются в облако. Работа с большими объемами данных и получение из них пользы – это задача, требующая комплексной обработки событий, аналитики и приемов машинного обучения;

- **безопасность**: при сведении всех элементов архитектуры воедино встают вопросы безопасности. Безопасность касается каждого компонента: от датчиков физических величин до ЦПУ и цифрового аппаратного обеспечения, систем радиосвязи и самих протоколов передачи данных. На каждом уровне необходимо обеспечить безопасность, достоверность и целостность. В этой цепи не должно быть слабых звеньев, поскольку интернет вещей станет главной мишенью для хакерских атак в мире.

Эта экосистема будет вовлекать в себя специалистов из различных технических областей, например, физиков, занимающихся разработкой приборов, проектирующих новые виды датчиков и многолетние батарейки. Инженеры-программисты встраиваемых систем, работающие над ведущими датчиками граничных устройств. Сетевые инженеры, умеющие работать с персональными сетями и глобальной вычислительной сетью, а также с программно-конфигурируемыми сетями. Специалисты по обработке данных, работающие над новейшими схемами машинного обучения на граничных устройствах и в облаке. DevOps-инженеры, которые успешно реализуют облачные решения различных масштабов, а также туманные вычисления. Интернет вещей всегда будет нуждаться в поставщиках услуг, например, компаниях, занимающихся реализацией проектов, системных интеграторах, поставщиках комплексных систем и изготовителях комплектного оборудования.

Интернет вещей против межмашинного взаимодействия

Предметом большинства вопросов и дискуссий на тему интернета вещей является следующее: что отличает IoT-мир от технологий межмашинного взаимодействия (M2M). Прежде чем интернет вещей стал популярной темой для разговоров, живейший интерес был направлен на межмашинное взаимодействие. Технологии межмашинного взаимодействия и интернета вещей очень схожи по своей сути, но есть существенное различие:

- **M2M**: это общая концепция, подразумевающая, что автономное устройство напрямую обменивается данными с другим автономным устройством. Под автономностью подразумевается способность узла инстанцировать и передать информацию другому узлу без участия человека. Форма передачи данных может быть разной. Очень часто бывает, что в устройстве M2M не интегрирован специализированный функционал для передачи информации. Это сразу отметает типичные устройства для выхода в интернет, которые регулярно применяются для работы с облачными сервисами и хранилищами. В M2M-системе обмен данными также может осуществляться без участия протокола IP – например, через последовательный порт или пользовательский протокол;
- **IoT**: IoT-системы могут содержать в себе несколько M2M-узлов (например, сеть Bluetooth Mesh, в которой реализован обмен данными без участия протокола IP), но данные агрегируются на граничном маршрутизаторе или шлюзе. Граничное устройство, такое как шлюз или маршрутизатор,

служит точкой входа в интернет. Или же некоторые датчики, отличающиеся более высокой вычислительной мощностью, могут пробиться на сетевой уровень интернета самостоятельно. Независимо от того, где происходит *выход* в интернет, отличительной чертой IoT-системы является то, что тем или иным образом она всегда подключена к интернету.

Благодаря перемещению данных с датчиков, граничных и умных устройств в интернет появляется возможность подсоединить простейшие устройства к устоявшемуся миру облачных сервисов. До того, как облачные технологии и мобильная связь стали частью повседневной реальности и перестали быть дорогостоящими, у простейших датчиков и встраиваемых систем отсутствовал способ за несколько секунд отправлять данные другим устройствам, хранить информацию сколь угодно долго и анализировать данные, чтобы выявлять тенденции и шаблоны. По мере развития облачных технологий беспроводные системы связи стали повсеместными, новые энергетические устройства, такие как литий-ионные аккумуляторы, обрели рентабельность, а модели машинного обучения нашли новые направления практического применения. Это сильно укрепило позиции интернета вещей. Если бы все эти технологии не сошлись воедино именно в тот самый момент, когда это произошло, мы до сих пор жили бы в мире M2M.

Полезность сети и законы Меткалфа и Бекстрома

По устоявшейся традиции полезность сети рассчитывается в соответствии с законом Меткалфа. В 1980 г. Роберт Меткалф сформулировал идею о том, что полезность любой сети пропорциональна квадрату численности пользователей системы. Когда речь идет об интернете вещей, в качестве пользователей можно рассматривать датчики или граничные устройства. В целом, закон Меткалфа выражается следующей формулой:

$$V \propto N^2,$$

где V – полезность сети; N – количество узлов в этой сети.

Чтобы лучше разобраться в этой формуле, посмотрим график на рис. 2.1, на котором также показана точка пересечения, которая отмечает порог, перейдя через который можно ожидать окупаемости инвестиций (положительное значение коэффициента ROI).

Недавно появился еще один пример, подтверждающий актуальность закона Меткалфа применительно к распределенным реестрам и поведению криптовалют. Мы подробнее разберем распределенные реестры в главе, посвященной вопросам безопасности.

В недавно вышедшем аналитическом докладе Кена Алаби говорится о том, что распределенные сети также подчиняются закону Меткалфа[1].

[1] Electronic Commerce Research and Applications. Т. 24, С (июль, 2017). С. 23–29.

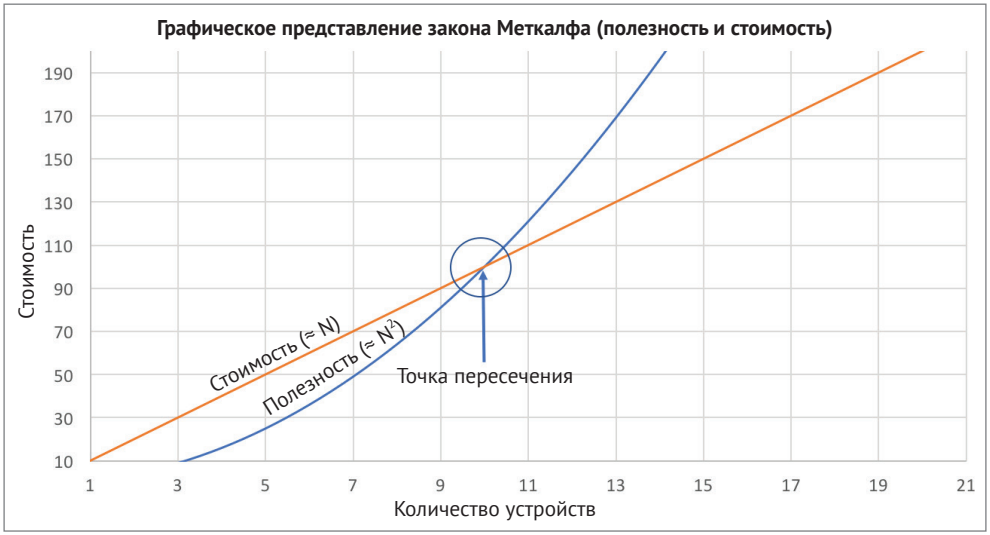


Рис. 2.1 ❖ Закон Меткалфа. Полезность сети в зависимости от показателя N. Стоимость каждого узла выражена формулой kN, где k – это произвольная постоянная. В данном примере значение k равно $10 за каждый граничный датчик IoT-сети. Важное значение – это точка пересечения, которая появляется в случае роста показателя полезности и обозначает тот момент, когда IoT-система достигает положительного значения коэффициента ROI

Закон Меткалфа не принимает в расчет ухудшение качества связи, которое может возникнуть в результате увеличения количества пользователей и/или объема данных на фоне сохранения изначальной пропускной способности сети. Также закон Меткалфа не принимает в расчет различие в уровнях сетевого обслуживания, ненадежную инфраструктуру (например, связь стандарта 4G LTE в движущемся транспортном средстве) или неблагоприятные факторы, влияющие на работу сети (например, DoS-атаки).

Чтобы сделать поправку на эти обстоятельства, мы применяем закон Бекстрома:

$$\sum_{i=1}^{n} V_{i,j} = \sum_{i=1}^{n} \sum_{k=1}^{m} \frac{B_{i,j,k} - C_{i,j,k}}{(1+r_k)^{t_k}},$$

где:

- $V_{i,j}$ обозначает полезность сети j для устройства i в данный момент времени;
- i – отдельный пользователь или устройство в сети;
- j – сама сеть;
- k – единичная транзакция;
- $B_{i,j,k}$ – преимущество, которое транзакция k дает устройству i в сети j;

- $C_{i,j,k}$ – стоимость транзакции k для устройства i в сети j;
- r_k – ставка дисконтирования на момент транзакции k;
- t_k – затраченное время (в годах) на транзакцию k;
- n – количество пользователей;
- m – количество транзакций.

Согласно закону Бекстрома, чтобы рассчитать полезность сети (например, IoT-решения), мы должны оценить все транзакции с каждого устройства и суммировать их полезность. Если сеть j по любой причине дает сбой, во сколько это обойдется пользователю? Это то влияние, которое оказывает IoT-сеть, и тут просматривается более сильная привязка к реальным показателям полезности. Сложнее всего в этом уравнении вычислить показатель B – преимущество транзакции. Если разбирать каждый IoT-датчик по отдельности, значение этого параметра может быть очень небольшим и несущественным (например, индикатор температуры какого-либо устройства не передавал данные в течение часа). В других случаях это значение может быть очень существенным (например, села батарейка датчика протечки воды, и складское помещение магазина оказалось затоплено, что повлекло за собой большие издержки и повышение стоимости страховки).

При разработке IoT-решений первое, что должен сделать архитектор – это понять, какую пользу принесет проект. В худшем случае IoT-система становится финансовой обузой и приносит клиенту только убытки.

Архитектура интернета вещей

IoT-архитектура, как мы уже упоминали, охватывает множество технологий. Каждый архитектор должен понимать, какое влияние выбранное проектное решение будет оказывать на всю систему в целом и каждую из её частей по отдельности. Сложности и многогранность интернета вещей связаны с тем, что эта технология гораздо более комплексная, чем традиционные технологии: её отличает не только большой размах, но и сочетание различных, часто не связанных между собой типов архитектуры. Количество возможных проектных решений поражает воображение. Например, на момент написания данной книги в мире существует более 700 IoT-провайдеров, предлагающих облачные хранилища, SaaS-компоненты, системы управления IoT, системы безопасности IoT и любые виды анализа данных. Добавьте сюда огромное количество различных протоколов персональных, локальных и глобальных сетей, которые постоянно меняются и корректируются в зависимости от региона. Выбор неподходящего протокола персональной сети может привести к проблемам с обменом данными и заметно низкому качеству сигнала, и ситуацию можно исправить, только добавив больше узлов в сеть. Архитектор должен учитывать интерференцию в локальных и глобальных сетях: каким образом данные снимаются с граничных устройств и передаются в интернет? Архитектор должен оценить отказоустойчивость системы и стоимость возможной потери данных. Какой уровень должен отвечать за отказоустойчивость си-

стемы – нижние уровни или уровень протокола? Архитектор также должен выбрать интернет-протоколы: MQTT или CoAP и AMQP, – а также необходимо продумать, как все это будет работать в случае перехода на другой облачный сервис. Также нужно решить, в какой точке будет осуществляться обработка данных. На этом этапе можно рассмотреть туманные вычисления как способ обработки данных рядом с источником, что решает проблему запаздывания и, что важнее, позволяет снизить загрузку сети и расходы при передаче данных по глобальным сетям и облачным сервисам. Далее мы рассматриваем все варианты анализа полученных данных. Неподходящий инструмент аналитики может стать причиной захламления системы избыточными данными или заставит вас пользоваться алгоритмами, которые требуют слишком больших вычислительных ресурсов для работы на граничных узлах. А как запросы, направляемые от облака к датчику, повлияют на работу батарейки самого датчика? Помимо всего этого широкого диапазона возможных вариантов, мы не должны забывать о системе безопасности, поскольку созданная нами IoT-система становится крупнейшей мишенью для атак в городе. Как вы видите, выбор огромен и каждое решение влияет на остальные. На данный момент нам доступно более 1,5 млн различных комбинаций архитектурных решений (рис. 2.2).

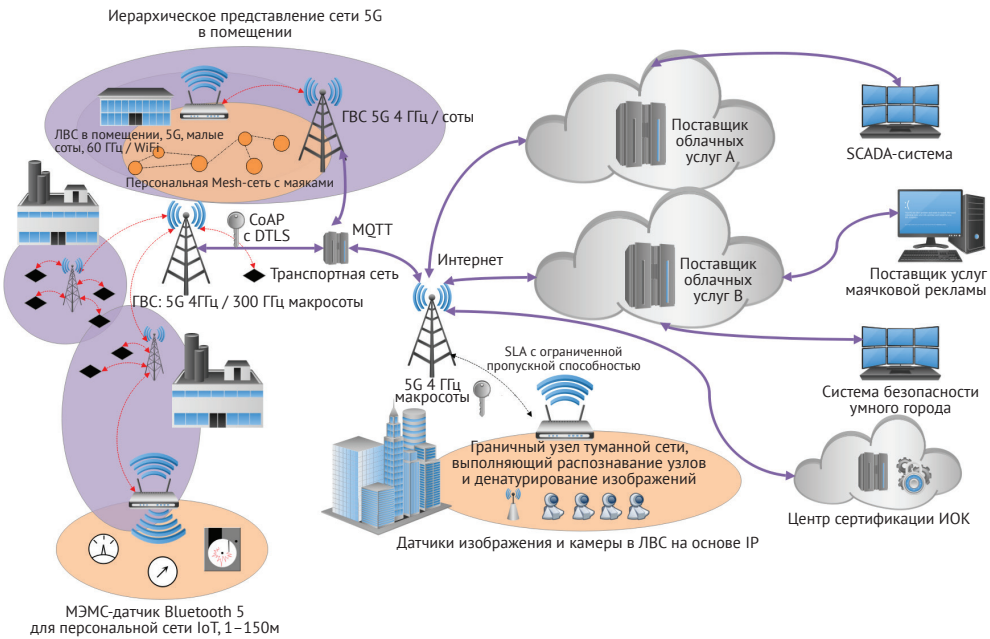


Рис. 2.2 ❖ Варианты IoT-решений. Полный спектр различных вариантов на всех уровнях IoT-архитектуры: от датчика до облака, и наоборот

Роль архитектора

Термин «архитектор» часто применяется в технических дисциплинах. Существуют архитекторы ПО, системные архитекторы и архитекторы решений. Даже в узких областях, таких как информатика и программирование, встречаются специальности, названия которых звучат как SaaS-архитектор, архитектор облачных решений, архитектор интеллектуальной обработки данных и пр. Это люди, которые отлично разбираются в указанных сферах, профессионалы своего дела. Эти вертикальные предметные области пересекают большое количество горизонтальных технологий. Данная книга предназначена для IoT-архитекторов. В книге мы берем горизонтальный срез, т. е. расскажем о нескольких из обозначенных выше предметных областей и объединим их в удобную, безопасную систему с большим потенциалом дальнейшего развития.

Мы погрузимся в теоретические аспекты, чтобы понять, как формируется IoT-система. В какие-то моменты нас будет интересовать только голая теория, например, теория информации и коммуникации. В других случаях мы будем говорить о вещах, которые тем или иным образом касаются IoT-систем или связаны с другими технологиями. Прочитав и осмыслив содержание данной книги, архитектор получит готовое руководство по различным составляющим интернета вещей, каждая из которых необходима для создания эффективной системы. Неважно, в чем вы лучше всего разбираетесь: будь то электроника, информатика или вы учились работать с архитектурой облачных систем, – эта книга поможет вам понять всю структуру в целом, что, по определению, и должен делать архитектор.

Эта книга предназначена также для широкого круга читателей. Каждая затронутая тема может быть полезна любителям и интересующимся, поскольку интернет вещей располагает всеми предпосылками к тому, чтобы разрастись до всемирного масштаба, распространить свое влияние на крупнейшие корпорации и включить в себя тысячи, миллионы граничных устройств.

Часть 1. Датчики и питание

Интернет начинается или заканчивается одним событием: простое движение, смена температуры или, может быть, рычаг защелкивает замок. В отличие от многих существующих ИТ-устройств, интернет вещей по большей части связан с физическим действием или событием. Он выдает реакцию на какой-то фактор реального мира. Иногда при этом один-единственный датчик может сгенерировать огромный объем данных, например, акустический датчик для профилактического осмотра оборудования. В иных случаях всего одного бита данных достаточно, чтобы передать жизненно важные сведения о состоянии здоровья пациента. Какой бы ни была ситуация, системы датчиков эволюционировали и, в соответствии с законом Мура, уменьшились до субнанометровых размеров и стали существенно дешевле. Именно к этому апеллируют

те, кто прогнозирует, что к интернету вещей будут подключены *миллиарды* устройств, и именно поэтому эти прогнозы оправдаются. Эта глава посвящена микроэлектромеханическим системам, датчикам и другим типам недорогих граничных устройств и их электрофизическим свойствам. Также в этой главе рассказывается о том, какие силовые и энергетические системы необходимы для питания этих граничных устройств. Нельзя считать, что граничные устройства снабжаются энергией по умолчанию. Миллиардам маленьких датчиков все равно нужен большой объем энергии. В этой книге мы еще будем возвращаться к теме электропитания, а также поговорим о том, как безобидные перемены в облачном сервисе могут кардинальным образом повлиять на всю энергетическую архитектуру системы в целом.

Часть 2. Передача данных

Большая часть этой книги посвящена установлению соединения и работе сетей. Бесчисленное множество других источников подробно рассказывают о разработке приложений, прогнозной аналитике и машинном обучении. В этой книге мы тоже обсудим эти темы, но не меньше внимания будет уделено процессу передачи данных. Интернета вещей не существовало бы без надежных технологий передачи данных из самых удаленных и неблагоприятных областей в крупнейшие центры сбора данных компаний Google, Amazon, Microsoft и IBM. Словосочетание «интернет вещей» содержит слово «*интернет*», поэтому мы должны изучить вопросы, касающиеся сетевых технологий, обмена данными и даже теории сигналов. Базовая опора интернета вещей – это не датчики и не приложения, а возможность установить соединение, как мы увидим в данной книге. Успешный архитектор понимает все сложности межсетевого взаимодействия датчика с глобальной вычислительной сетью и наоборот (взаимодействия ГВС с датчиком).

Раздел про передачу данных и сетевое соединение начинается с теоретических аспектов и математических основ коммуникации и работы с информацией. Успешным архитекторам понадобятся тренировочные инструменты и модели – не столько для того, чтобы разобраться, почему определенные протоколы не очень эффективны, сколько для того, чтобы спроектировать будущие системы, которые способны расширяться до необходимых для интернета вещей масштабов. Эти инструменты включают в себя динамические характеристики радиосигнала, такие как анализ спектра и электропитания, отношение сигнал/шум, потери в тракте передачи и интерференция. В этой части также подробно расписаны основы теории информации и ограничения, которые влияют на общую пропускную способность и качество данных. Кроме того, освещаются основные положения закона Шеннона. Спектр сигнала беспроводной связи также не безграничен и распределяется между несколькими устройствами, а архитектор, разрабатывающий широкомасштабную IoT-систему, должен понимать, как и каким образом распределяется спектр. Теоретические

основы и практики, рассматриваемые в данной части, пригодятся при чтении остальных разделов книги.

Передача данных и установление сетевого соединения будут происходить на базе систем связи ближнего действия – персональных сетей (PAN), обычно построенных без соблюдения правил IP-протокола. Глава, посвященная персональным сетям, включает в себя информацию о новом протоколе Bluetooth 5 и mesh-сети, а также подробно рассказывает о протоколах Zigbee и Z-Wave. Это яркий пример многообразия беспроводных систем связи IoT. Далее мы рассмотрим беспроводные локальные сети и системы связи на основе IP-протокола, включая широкий диапазон Wi-Fi-сетей на основе стандартов IEEE 802.11, 6LoWPAN и технологии Thread. В главе также говорится о новых стандартах Wi-Fi-связи, например, 802.11p для передачи информации между транспортными средствами.

Этот раздел завершается рассказом о дальней связи на основе сотовых стандартов (4G LTE) и подробным описанием принципов действия и инфраструктуры, стоящей за стандартом 4G LTE и новыми стандартами, обеспечивающими работу интернета вещей и межмашинное взаимодействие, такими как Cat-1 и Cat-NB. В этой главе также много внимания уделяется наиболее перспективным свойствам находящегося на стадии разработки стандарта 5G, чтобы подготовить архитектора к планируемому увеличению радиуса действия сигнала, а также к тому, что каждое устройство будет использовать узкий диапазон частот. Также в этом разделе рассказывается о проприетарных протоколах LoRaWAN и Sigfox, чтобы показать совершенно иной тип архитектуры.

Часть 3. Интернет-маршрутизация и протоколы

Для передачи данных от датчиков в интернет-пространство необходимы две технологии: маршрутизатор-шлюз и опорные интернет-протоколы, обеспечивающие эффективность обмена данными. В данном разделе описана роль технологий маршрутизации на границах сети и рассказано, как эти технологии связывают датчики персональной вычислительной сети с интернетом. Маршрутизатор особенно важен в таких аспектах, как безопасность, управление и направление данных. Граничные маршрутизаторы управляют и следят за состоянием соответствующих mesh-сетей, а также выравнивают и поддерживают качество данных. Также огромное значение принадлежит конфиденциальности и безопасности данных. В этой части разъясняется роль маршрутизатора в создании виртуальных частных сетей, виртуальных локальных сетей и программно-определяемых глобальных сетей. Они в буквальном смысле могут содержать тысячи узлов, обслуживаемых единственным граничным маршрутизатором, и в какой-то степени маршрутизатор служит расширением для облака, как мы увидим в главе 10.

В этой части книги продолжена тема протоколов, необходимых для обмена данными между узлами, маршрутизаторами и облачными сервисами в преде-

лах IoT-системы. Интернет вещей открыл дорогу новым IoT-протоколам, которые выходят на один уровень с традиционными протоколами HTTP и SNMP, применяющимися уже несколько десятков лет. Для передачи IoT-данных требуются эффективные, энергосберегающие протоколы с малой задержкой, способные легко и безопасно отправлять данные в облако и из него. В этой главе рассказывается и о таких протоколах, как вездесущий MQTT, AMPQ и CoAP. Чтобы показать, как они применяются и в чем их сильные стороны, приводятся примеры.

Часть 4. Туманные и граничные вычисления, аналитика и машинное обучение

На этом этапе необходимо решить, что делать с потоком данных, поступающих в облачный сервис из граничного узла. Для начала мы поговорим о различных аспектах архитектуры облачных систем (модели SaaS, IaaS и PaaS). Архитектор должен разбираться в том, что такое поток данных, и в типичных схемах построения облачных сервисов (что они собой представляют и как применяются). В качестве модели облачной системы мы берем проект OpenStack и рассматриваем различные составляющие: от модулей Ingestor до озер данных и инструментов аналитики. Чтобы научиться верно оценивать, как система будет развиваться и расти, необходимо разобраться во всех тонкостях и сложностях архитектуры облачных систем. Архитектор также должен понимать, какое влияние на IoT-систему оказывает запаздывание. Кроме того, не все нужно отправлять в облако. Пересылка всех IoT-данных обходится значительно дороже, чем их обработка на границе сети (граничные вычисления) или включение граничного маршрутизатора в зону, которую обслуживает облачный сервис (туманные вычисления). В этом разделе подробно освещаются новые стандарты туманных вычислений, такие как архитектура OpenFog.

Данные, которые были получены путем преобразования аналогового физического воздействия в цифровой сигнал, могут иметь большой вес. Именно здесь в игру вступают средства аналитики и процессоры правил IoT-системы. Степень сложности ввода в действие IoT-системы зависит от того, какое решение проектируется. В некоторых ситуациях все довольно просто: например, когда на граничный маршрутизатор, контролирующий несколько датчиков, нужно установить простой процессор правил, отслеживающий аномальные скачки температуры. Другая ситуация – огромное количество структурированных и неструктурированных данных в режиме реального времени передается в облачное озеро данных, что требует высокой скорости обработки (для прогнозной аналитики) и долгосрочного прогнозирования на базе высокотехнологичных моделей машинного обучения, таких как рекуррентная нейронная сеть в пакете анализа сигналов с корреляцией по времени. В этой главе подробно рассказывается о примерах и сложностях аналитики: от сложных обработчиков событий до байесовских сетей и формирования нейронных сетей.

Часть 5. Угроза и безопасность в интернете вещей

Завершает книгу раздел, посвященный уязвимостям IoT и хакерским атакам. Многие IoT-системы не будут ограничены безопасным пространством дома или офиса. Они будут располагаться в общественных местах, в очень отдаленных областях, в движущихся транспортных средствах или даже внутри человека. Интернет вещей – это огромная единая мишень для любых видов хакерских атак. Мы были свидетелями бесконечного множества направленных на IoT-устройства учебных атак, хорошо организованных взломов и даже брешей в системе безопасности национального масштаба. В этой главе подробно расписаны некоторые особенности таких уязвимостей и способы их устранения, которые каждый архитектор должен взять на вооружение, если хочет поставить *на рельсы* персональную или корпоративную IoT-систему. Мы рассказываем о законопроекте, предложенном в Конгрессе США в качестве меры обеспечения безопасности интернета вещей, и пытаемся разобраться в побудительных мотивах и возможных последствиях подобного шага со стороны правительства. В этой части также перечислены стандартные мероприятия, направленные на защиту интернета вещей или любого компонента сети. Также мы подробно поговорим о новых технологиях, таких как распределенные реестры и программно-определяемый периметр, и расскажем, какие технологии обеспечения безопасности IoT появятся в будущем.

Заключение

Эта книга познакомит вас с разнообразием технологий, лежащих в основе интернета вещей. Мы кратко рассказали обо всех сферах и темах, обозначенных в данной главе. Архитектор должен понимать, как на стыке этих совершенно различных технических дисциплин создается надежная, оптимизированная система с потенциалом расширения. Кроме того, архитектор должен уметь доказать, что IoT-система приносит дополнительное преимущество конечному пользователю или клиенту. С этой задачей помогут справиться законы Меткалфа и Бекстрома, о которых также можно прочитать в данной книге.

Начиная со следующей главы, мы погрузимся в архитектуру интернета вещей и поговорим обо всем, от датчика до облака.

Глава 3

Датчики, оконечные точки и системы питания

Интернет вещей (**IoT**) начинается с источников данных или исполнительных устройств. Это называется **оконечными точками**, и, имея выход в интернет, они могут быть объединены в единую сеть. При обсуждении IoT в целом, рассмотрение фактических источников данных часто игнорируется. Что такое эти источники? Это датчики, а данные, которые они предоставляют, образуют распределенные по времени потоки данных. Для таких потоков необходимо обеспечить возможность передачи, анализа и сохранности информации. Ценность IoT в том, что это комплексное решение, а данные, предоставляемые датчиком, играют в этом комплексе ключевую роль. Таким образом, проектировщику совершенно необходимо понимать, что это за данные и как их правильно интерпретировать. Помимо понимания того, какие данные собираются и как они образуются в массиве IoT, полезно знать, что именно и в каких пределах измеряется. Например, система должна учитывать устройства, потерявшие связь, и ошибочные данные. Проектировщик должен понимать причины, по которым данные, полученные от датчиков, могут быть ненадежными, а также причины, вследствие которых полевой датчик может выйти из строя. По сути, мы соединяем аналоговый мир с цифровым. Большинство аналоговых устройств, интегрируемых в цифровое пространство, это датчики, поэтому важно понимать их роль и значение.

В двух словах о том, что такое IoT. Количество датчиков и исполнительных устройств, объединенных в единую сеть, значительно выросло, поэтому проектировщику чрезвычайно важно понимать их взаимодействия. В этой главе будут рассмотрены и систематизированы электронные сенсорные устройства, так как важно понимать принципы того, что измеряется и как. Каждый должен задаться вопросом: «Какой тип датчика или оконечного устройства следует использовать для решения проблемы, которая стоит передо мной?» При развертывании IoT необходимо учитывать множество аспектов: стоимость, опциональность, размеры, продолжительность безаварийной работы и точность измерений. Кроме того, в литературе по IoT мощность и энергия, потребляемые периферийными устройствами, рассматриваются редко, но именно эти показатели имеют решающее значение при создании надежных и долговечных

технологий. Эта глава должна помочь читателю достичь четкого понимания сенсорных технологий и их ограничений.

В этой главе мы рассмотрим следующие темы:

- сенсорные устройства – от термопар до датчиков MEMS для систем видеонаблюдения;
- системы генерации энергии;
- системы хранения энергии.

Сенсорные устройства

Для начала рассмотрим сенсорные, или, иначе, входные устройства. Это могут быть устройства самых разных типов и сложности: от простых термопар до продвинутых видеосистем. В этом разделе представлен широкий спектр сенсоров: именно они имеются в виду, когда говорят о «миллиардах вещей IoT». Одной из причин взрывного роста IoT является тот факт, что эти сенсорные системы, благодаря достижениям полупроводниковой промышленности и микроэлектроники, приобрели малые физические размеры и значительно подешевели.

Термопары и температурные датчики

Датчики температуры – это наиболее распространенный тип датчиков. Они применяются повсеместно: от интеллектуальных термостатов для холодных складов до охладителей промышленного оборудования, и, скорее всего, это первый сенсор, с которым вы столкнетесь в IoT.

Термопары

Термопара (или TC) – это устройство для измерения температуры, которому не требуется источник питания, потому что оно само генерирует сигнал малой амплитуды (обычно микровольты). Термопара – это два проводника, изготовленные из двух разных материалов, соединенные в точке измерения температуры. На металлическом электроде, в зависимости от его температуры, возникает электрический потенциал. У разных металлов уровень этого потенциала различен. Этот эффект известен как **электродвижущий эффект Зеебека**, его суть состоит в том, что разность потенциалов между двумя различными металлами находится в нелинейной зависимости от их температуры.

Величина напряжения зависит от свойств выбранного металла. Крайне важно, чтобы концы проводов были термически изолированы от системы (и провода должны иметь одинаковую контролируемую температуру). На рис. 3.1 приведена блок-схема измерения температуры с помощью термопары. Для повышения точности измерений разность потенциалов, индуцируемая термопарой, обычно измеряется методом, который называется **методом компенсации**.

Поскольку разным температурам соответствуют разные уровни напряжения, а зависимость между измеряемой температурой и индуцируемым напряжением нелинейная, для перевода измеренного потенциала в температуру, как правило, используется справочная таблица.

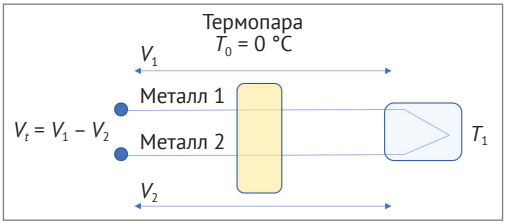


Рис. 3.1. ❖ Блок-схема измерения температуры с помощью термопары

Термопары следует использовать при выполнении не слишком ответственных измерений, поскольку показания отдельных термопар, при прочих равных условиях, могут разниться. Это вызвано тем, что различные тонкие примеси, входящие в состав измерительных электродов, могут приводить к несоответствиям со справочными таблицами. Можно, конечно, воспользоваться высокоточными (прецизионными) термопарами, но они и стоить будут, соответственно, дороже. Другим эффектом, влияющим на точность измерений, является старение. Так как термопары часто используются в промышленных условиях, высокотемпературные среды с течением времени могут ухудшать точность датчиков. Поэтому IoT-решения должны учитывать изменения, происходящие с датчиками в процессе их эксплуатации.

Термопары хорошо работают в широком диапазоне температур. Для различных комбинаций металлов приняты цветовая и буквенная маркировки, указывающие тип термопары (например, E, M, PT-PD). Обычно подобные датчики используются в промышленных и высокотемпературных средах при проведении измерений в местах, удаленных от оператора.

На рис. 3.2 приведена зависимость ЭДС от температуры для нескольких типов термопар.

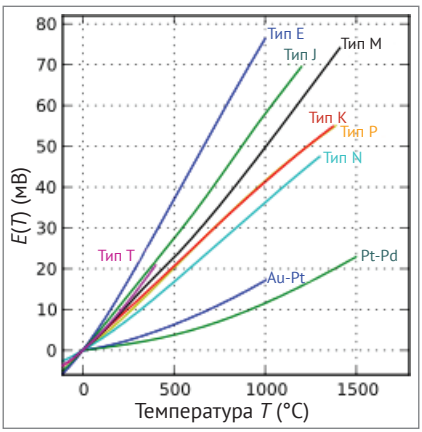


Рис. 3.2. ❖ Зависимость ЭДС термопары от ее температуры $E(T):T$

Резистивные датчики температуры

Резистивные датчики температуры (Resistance Temperature Detectors – RTD) работают в узком диапазоне температур (ниже 600 °C), но позволяют выполнять измерения с большей точностью, чем термопары. Обычно они изготавливаются из очень тонкой платиновой проволоки, плотно намотанной на керамический или стеклянный сердечник. Электрическое сопротивление такой конструкции пропорционально ее температуре. Поскольку в основе измерений лежит измерение сопротивления, для работы с RTD необходим внешний источник питания с выходной силой тока 1 мА.

RTD изготавливаются в соответствии с принятыми стандартами, в которых определены допустимый диапазон и шаг измерений. Например, для датчика 200 PT100 RTD шаг измерений составляет 0,00200 Ом/°C, а диапазон измерений лежит в пределах от 0 до 100 °C. В пределах этого диапазона зависимость сопротивления RTD от его температуры сохраняет линейный характер. В соответствии со стандартами RTD выпускаются в двух-, трех- и четырехпроводном исполнении, четырехпроводные модели используются исключительно в системах высокоточной калибровки. Для увеличения разрешения измерений RTD часто используют в мостовых схемах, при этом обычно показания линеаризуются программно (рис. 3.3).

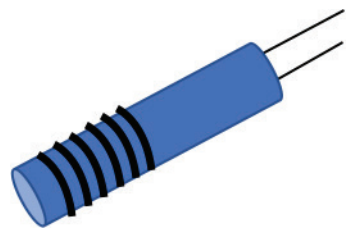


Рис. 3.3 ❖ Недостатки проволочных RTD

RTD редко используются в диапазоне выше 600 °C, что ограничивает их применение в промышленности. При высоких температурах платина может загрязняться, что приводит к ошибочным показаниям, однако при измерениях в пределах заданного диапазона, RTD демонстрируют достаточно точные и стабильные результаты.

Термисторы

Наконец **термистор** – это тоже датчик температуры, электрическое сопротивление которого зависит от его температуры. Этот тип датчиков обеспечивает более высокую точность измерений в сравнении с RTD. По сути, это терморезисторы, но с очень нелинейной зависимостью сопротивления от температуры. Их часто используют в качестве сглаживающих фильтров, для ограничения скачков тока, а также в случаях, когда необходима высокая степень разрешения измерений в узком диапазоне температур. Существует два типа термисто-

ров: NTC (их сопротивление уменьшается при повышении температуры) и PTC (их сопротивление возрастает с повышением температуры). Основное отличие от RTD заключается в том, что термисторы изготавливаются из керамики или полимеров, тогда как основой RTD всегда является металл.

Термисторы находят применение в медицинском и научном оборудовании, в пищевой промышленности, инкубаторах и в таких бытовых приборах, как термостаты.

Сводная таблица датчиков температуры

В табл. 3.1 перечислены типы датчиков температуры, приведены примеры их использования, а также указаны преимущества использования конкретных датчиков.

Таблица 3.1. Сводная таблица датчиков температуры

Категория	Термопара	Резистивные датчики температуры	Термистор
Температурный диапазон (в °C)	От –180 до 2,320	От –200 до 500	От –90 до 130
Время реакции	Быстро (микросекунды)	Медленно (секунды)	Медленно (секунды)
Размеры	Большие (~10 мм)	Небольшие (~ 5 мм)	Небольшие (~ 5 мм)
Точность	Низкая	Средняя	Очень высокая

Эффект Холла и датчики тока

Датчик Холла – это полоска металла, через которую пропущен электрический ток. Поток заряженных частиц, проходящих через магнитное поле, отклоняется от прямолинейного направления. Если направление магнитного поля перпендикулярно плоскому проводнику, то на его противоположных сторонах будет возникать разность потенциалов, обусловленная тем, что разноименно заряженные частицы будут собираться на противоположных его сторонах. Таким образом, одна сторона плоского проводника окажется заряжена положительно, а другая отрицательно, и возникнет разность потенциалов. Такая разность потенциалов называется **напряжением Холла**, а сам эффект возникновения этого напряжения называется **эффектом Холла**. Это проиллюстрировано на рис. 3.4, приведенном ниже. Когда через металлическую полосу (как показано на рис. 3.4), помещенную в магнитное поле, проходит ток, электроны притягиваются к одной ее стороне, а дырки – к другой (см. кривую на рис. 3.4). Такое расслоение порождает электрическое поле, которое можно измерить. Если поле достаточно сильное, оно нейтрализует действие магнитного поля, и носители заряда сохраняют прямолинейное движение:

Эффект Холла применяется в датчиках тока, для измерения переменного и постоянного тока. Существует два типа таких датчиков: с разомкнутым и замкнутым контуром. Датчики с замкнутым контуром более дорогие, их часто используют в схемах с питанием от батарей.

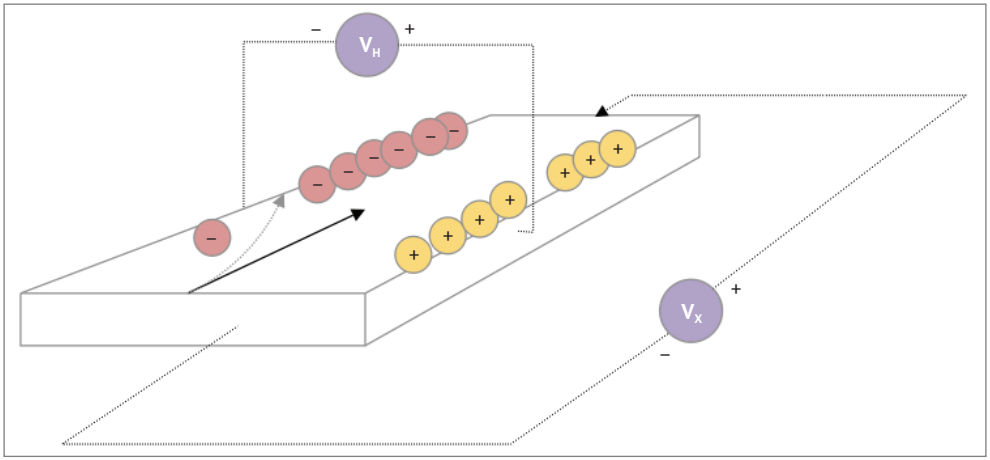


Рис. 3.4 ❖ Иллюстрация эффекта Холла

Типичная область применения датчиков Холла: датчики положения, магнитометры, высоконадежные переключатели и указатели уровня воды. Они используются также в промышленных датчиках для измерения скорости вращения различных узлов и механизмов. Кроме того, что эти датчики недороги в изготовлении, они не требовательны к условиям эксплуатации и могут работать в самых суровых условиях.

Фотоэлектрические датчики

Датчики для обнаружения света или определения его интенсивности используются во многих устройствах IoT. Такие устройства необходимы, например, в системах безопасности, интеллектуальных коммутаторах или в системах управления уличным освещением. Существует два типа таких датчиков, принцип действия которых понятен из их названия. Фоторезистор изменяет сопротивление в зависимости от интенсивности света, а фотодиод преобразует свет в электрический ток.

Фоторезисторы изготавливаются из полупроводников с высоким сопротивлением. Их сопротивление уменьшается при увеличении интенсивности освещения. В темноте сопротивление фоторезистора может иметь довольно высокое сопротивление (порядка мегаом). Фотоны, поглощаемые полупроводником, переводят электроны в зону проводимости, тем самым увеличивая проводимость материала. Фоторезисторы чувствительны к длине волны падающего света, поэтому их типов и модификаций существует огромное множество. А вот фотодиоды – это полноценные полупроводниковые приборы с p-n-переходом. Такие устройства реагируют на свет, создавая электронно-дырочную пару. Поток дырок, движущихся к аноду, и электронов, движущихся к катоду, создает электрический ток. Таким образом работают традиционные солнечные батареи, производящие электричество под воздействием солнеч-

ных лучей. Если на фотодиод подать обратное напряжение, то можно регулировать его чувствительность или время отклика (см. табл. 3.2).

Таблица 3.2. Фотоэлектрические датчики

Категория	Фоторезистор	Фотодиод
Светочувствительность	Низкая	Высокая
Активный/Пассивный (полупроводниковый)	Пассивный	Активный
Чувствительность к температуре	Высокая чувствительность	Низкая
Время реакции на изменение освещенности	Длительное (от 10 мс до 1 с)	Короткое

Датчики PIR

Пироэлектрический инфракрасный (Pyroelectric Infrared – PIR) датчик состоит из двух слотов, заполненных материалом, реагирующим на инфракрасное излучение и тепло. Типичные варианты применения таких датчиков – это тепловые датчики движения систем безопасности. Обычно такие датчики оснащаются линзой Френеля, посредством которой формируется зона обнаружения. Такая зона имеет форму арки, раскрывающейся наружу. Когда теплое тело входит или, наоборот, покидает зону обнаружения, чувствительные элементы формируют электрический сигнал. В ПИР-датчиках используются кристаллические материалы, которые способны генерировать электрический ток под воздействием ИК-излучения. Все вместе это образует так называемый **полевой транзистор** (Field Effect Transistor – FET), который фиксирует изменение тока и посылает сигнал на усилительное устройство. ПИР-датчики хорошо работают в диапазоне волн от 8 до 14 мкм, этот диапазон характерен для излучений человеческого тела.

На рис. 3.5 ниже изображены два элемента PIR, формирующие две зоны обнаружения. Это позволяет не только отслеживать все пространство комнаты, но и определять направление перемещений.

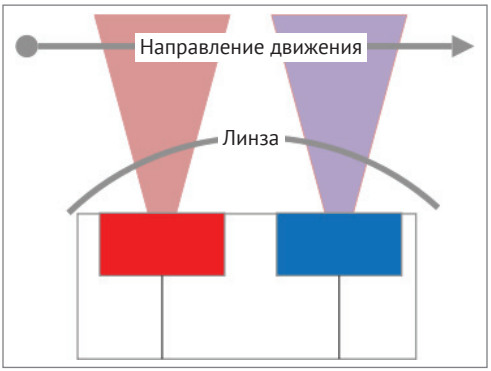


Рис. 3.5 ❖ Датчик PIR. Два элемента реагируют на источник ИК, движущийся в зоне обнаружения

Для сканирования большей площади с помощью одного датчика потребуется несколько линз Френеля, которые будут фокусировать на PIR изображение отдельных областей отслеживаемой территории. Такая фокусировка обеспечивает концентрацию инфракрасного излучения непосредственно в области FET. Как правило, такие устройства позволяют проектировщику контролировать чувствительность (диапазон) и время реакции.

Время реакции указывает, через какой промежуток времени будет послан сигнал после обнаружения движения на охраняемой территории. Чем меньше время реакции, тем больше событий может быть зафиксировано. На рис. 3.6 приведена диаграмма типичного ПИР-датчика, оснащенного линзой Френеля с фиксированным фокусным расстоянием и сфокусированной на подложку.

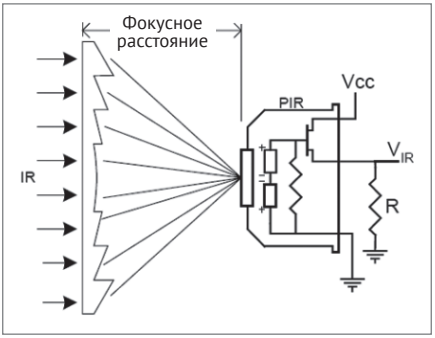


Рис. 3.6 ❖ Слева: линза Френеля фокусирует ИК-излучение на ПИР-датчик. Запись по применению Cypress Microsystems AN2105

LiDAR и активные датчики

До сих пор мы рассматривали пассивные датчики, которые просто реагируют на изменения окружающей среды. В этом разделе будут рассмотрены активные датчики. Активное зондирование включает в себя посылку сигнала и его анализ после возвращения. Такие датчики позволяют судить об окружающей среде не только качественно (зафиксировать факт движения), но и количественно (измерить скорость движения). Эта область весьма обширна, поэтому мы сосредоточимся на LiDAR-датчиках, являющихся основой для активных зондирующих систем.

Датчики светового обнаружения и измерения (Light Detecting and Ranging – LiDAR). Этот тип датчика измеряет расстояние до цели путем измерения отраженных от объекта лазерных импульсов. Если датчик PIR лишь обнаружит движение в пределах своего диапазона, LiDAR способен измерять количественные характеристики этого движения. Впервые такой датчик был продемонстрирован в 1960-х гг., а в настоящее время широко используется в сельском хозяйстве, автоматизированных и самодвижущихся транспортных средствах, робо-

тотехнике, при наблюдениях и исследованиях окружающей среды. Этот тип активных измерительных устройств способен анализировать объекты любого типа. Они используются для анализа газов, атмосферы, облачных образований и композиций, частиц, скорости движущихся объектов и т. д.

LiDAR – активная сенсорная технология, построенная на основе лазера. Когда лазерный луч падает в объект, какая-то его часть отражается и возвращается обратно к излучателю LiDAR. Используемые лазеры обычно имеют длину волны от 600 до 1000 нм и относительно недороги. Их мощность ограничена по соображениям безопасности, чтобы предотвратить повреждение глаз. Некоторые датчики LiDAR работают в диапазоне 1550 нм, поскольку эта длина волны не воспринимается человеческим глазом, что делает их безвредными даже при высокой интенсивности. Системы LiDAR способны сканировать очень большие пространства и могут работать даже со спутников. Такая система посылает лазерные импульсы с частотой до 150 000 импульсов в секунду и фиксирует их отражение массивом фотодиодов. Иногда прохождение посылаемых и отраженных лазерных импульсов регулируется системой вращающихся зеркал, что формирует трехмерное изображение окружающей среды. Для каждого передаваемого лучевого импульса фиксируется угол отражения, измеряется **время пролета** (Time of Flight – TOF) и местоположение GPS. Эти параметры позволяют получить полное представление об исследуемом пространстве.

Уравнение для расчета расстояния до наблюдаемого объекта относительно просто:

$$\text{Расстояние} = \frac{(\text{Скорость пролета} \times \text{Время пролета})}{2}.$$

Другие активные датчики работают по тому же принципу, что и LiDAR. Каждый из них посылает какой-либо сигнал, который, отражаясь, возвращается к датчику, что и создает изображение наблюдаемой сцены или указывает на то, что произошло какое-либо событие. Эти датчики намного сложнее, чем простые пассивные датчики, потребляют больше энергии, стоят дороже и требуют большего пространства (рис. 3.7).

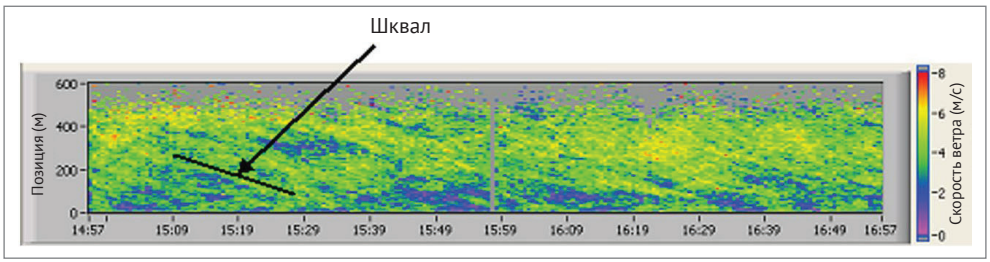


Рис. 3.7 ❖ LiDAR: пример изображения, полученного с помощью LiDAR. Такие картинки используемого для анализа атмосферных порывов ветра, для обеспечения защиты ветровых турбин. Изображение предоставлено NASA

Датчики MEMS

Промышленное производство **микроэлектромеханических систем** (Microelectromechanical systems – MEMS) началось в 1980-х гг., но впервые они появились еще в 1960-х гг., когда компанией Kulite Semiconductor был представлен пьезорезисторный датчик давления. По сути, это миниатюрные механические структуры, которые взаимодействуют с электронным блоком управления. Как правило, размеры таких датчиков лежат в диапазоне от 1 до 100 мкм. В отличие от других датчиков, упомянутых в этой главе, механические структуры MEMS могут вращаться, растягиваться, изгибаться, изменять форму, что и вызывает изменения электрического сигнала. Этот сигнал, полученный от отдельного конкретного датчика, фиксируется и измеряется.

Процесс изготовления MEMS-устройств типичен, для производства полупроводниковых приборов. На кристалл кремния наносится многослойная маска, далее следуют операции литографии, осаждения и травления. Затем матрица MEMS дополняется другими элементами, такими как операционные усилители, аналого-цифровые преобразователи и прочие вспомогательные схемы. Как правило, устройства MEMS имеют относительно большие размеры от 1 до 100 микрон, тогда как типичные кремниевые структуры имеют размеры 28 нм или меньше. Поэтому изготовление MEMS производится послойно, последовательное травление различных слоев создает устройство с трехмерной структурой.

Помимо применения в сенсорных системах, устройства MEMS можно встретить, например, в печатающих головках струйных принтеров или в современных проекторах, таких как **цифровые светопропускающие экраны** (digital light processor – DLP). Возможность создания чувствительных MEMS-устройств размером с булавочную головку, позволяет увеличить количество объектов IoT до миллиардов.

MEMS-акселерометры и гироскопы

Акселерометры и гироскопы широко применяются в различных мобильных устройствах для позиционирования и отслеживания движения, в таких, например, как шагомеры и фитнес-трекеры. Эти устройства используют пьезоэлектрический элемент MEMS, на котором в ответ на движение возникает ЭДС. Гироскопы обнаруживают вращательное движение, а акселерометры реагируют на изменение линейного движения. Диаграмма на рис. 3.8 иллюстрирует принцип работы акселерометра. Обычно массивное тело, закрепленное в определенной точке, при возникновении ускорения через пружину создает механическое напряжение в MEMS, изменяя его электрическую емкость. Величина ускорения определяется при измерении емкости в цепи MEMS. Может показаться, что такой акселерометр будет реагировать на ускорение лишь в одном направлении.

Но, как показано на рисунке ниже, такой акселерометр будет отвечать на ускорение в любом из направлений координатных осей (X, Y, Z).

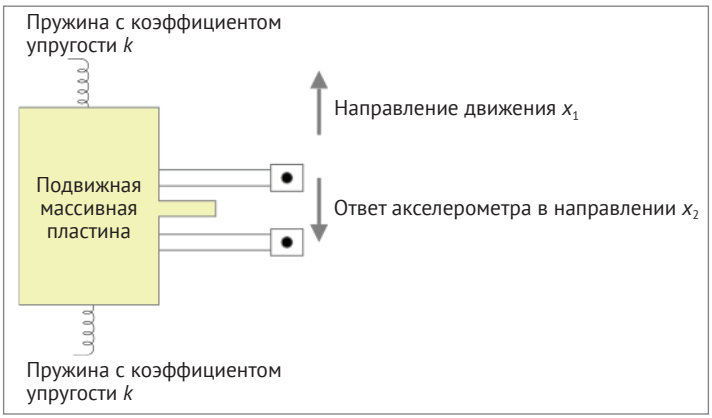


Рис. 3.8 ❖ Акселерометр: принцип измерения ускорения с использованием массивного тела, подвешенного на пружине. Как правило, используется для измерения ускорения в направлении координатных осей

Принцип действия гироскопов основан на эффекте Кориолиса для вращающейся системы отсчета. На рис. 3.9 ниже представлена суть этого эффекта. Объект, помещенный на вращающийся диск, двигаясь равномерно вследствие вращения самого диска, начинает двигаться по дуге. Таким образом, для сохранения прямолинейного движения и достижения северной точки диска объекту требуется дополнительное ускорение.

Это ускорение Кориолиса. В устройстве MEMS нет вращающегося диска, зато имеется резонансная частота, приложенная к последовательности концентрических колец, устроенных на кремниевой подложке в процессе изготовления MEMS.

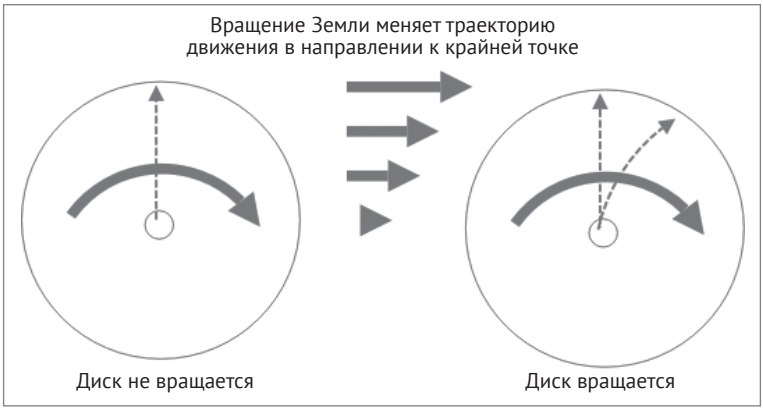


Рис. 3.9 ❖ Акселерометр: влияние вращения диска на путь, совершаемый при движении к северу

Эти концентричные кольца разрезаны на маленькие дуги. Концентрические кольца позволяют точно оценивать вращательное движение в широком диапазоне. Одиночные кольца необходимо устраивать на жестких опорных балках, и они не столь надежны. При разбивке колец на дуги структура теряет жесткость и становится более чувствительна к вращению. Источник постоянного тока через электроды, прикрепленные к кольцам, создает электростатическое поле. Это поле вызывает резонансные колебания системы колец. Если фиксируется изменение в таких резонансных колебаниях колец, значит, присутствует ускорение Кориолиса, которое определяется следующим уравнением:

$$a = -2\omega \times v.$$

Это уравнение показывает, что ускорение, вызванное вращением системы, пропорционально угловой скорости вращающегося диска, как показано на рис. 3.9, или резонансной частоте MEMS-устройства, как показано на рис. 3.10. Сила Кориолиса изменяет зазор между кольцами, установленный статическим полем источника постоянного тока, и, как следствие, общую емкость системы. Внешние электроды фиксируют отклонение в кольце, а внутренние обеспечивают измерения емкости.

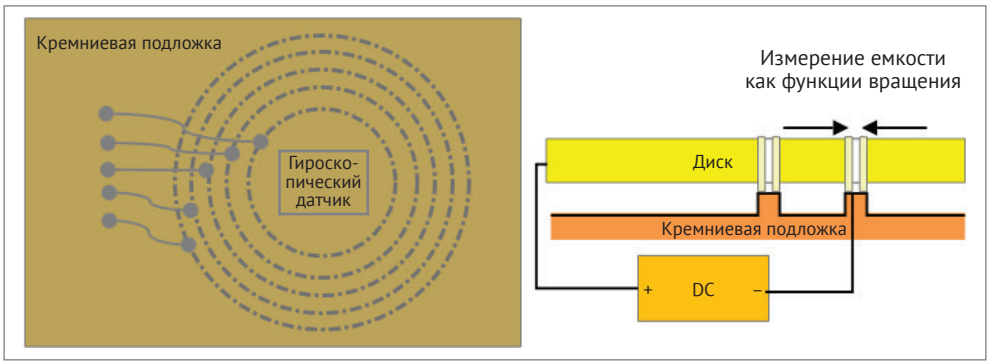


Рис. 3.10 ❖ Справа: концентрические разрезанные кольца, устроенные на кремниевой подложке, образуют гироскопический датчик. Слева: дисковые промежутки соединенные соответствующим образом

Для работы как гироскопов, так и акселерометров требуется источник питания и операционный усилитель для формирования сигнала, пригодного для обработки цифровым процессором.

Подобные устройства могут быть изготовлены в очень маленьких размерах. Например, устройство InvenSense MPU-6050 содержит 6-осевой гироскоп и акселерометр в корпусе размером 4×4×1 мм. Оно потребляет ток в 3,9 мА и, следовательно, является датчиком с низким энергопотреблением.

Микрофоны MEMS

MEMS-устройства могут также использоваться для фиксации звука и вибрации. Этот тип MEMS-устройств имеет непосредственное отношение к рассмотренным ранее акселерометрам. При развертывании систем IoT фиксация звука и вибрации широко применяется в их промышленных приложениях для профилактического и прогностического обслуживания. Например, в химической промышленности аппарат, который вращает загружаемые реактивы для их перемешивания, или центрифуги должны быть строго горизонтально ориентированы. А звуковой или вибрационный MEMS-блок обычно используется для контроля за исправностью и безопасностью такого оборудования.

Этому типу датчиков требуется аналого-цифровой преобразователь с достаточно высокой частотой дискретизации. Необходим также усилитель выходного сигнала. Импеданс MEMS-микрофона составляет порядка нескольких сотен ом (что предъявляет особые требования к используемому усилителю). Микрофон MEMS может быть аналоговым или цифровым. Аналоговый микрофон подключается к источнику постоянного тока и аналого-цифровому преобразователю. Цифровой микрофон имеет встроенный АЦП в непосредственной близости от детектора звука. Это является преимуществом при наличии помех от сотовых телефонов или Wi-Fi-устройств, которые могут повлиять на работу АЦП.

Для передачи выходного сигнала цифрового MEMS-микрофона используется два типа модуляции: **модуляция плотности импульсов** (pulse density modulated – PDM) и модуляция I^2S. PDM – это протокол с высокой частотой дискретизации и двумя каналами приема-передачи (левый и правый каналы). Прием-передача сигнала и тактовой частоты происходит по отдельным линиям, а прием-передача сигнала в левый и правый каналы разделена во времени (чередуется) в соответствии с тактовым сигналом. Протокол I^2S не обладает высокой частотой дискретизации, но при передаче сигнала звуковой частоты (в диапазоне от Гц до kHz) работает с приемлемым уровнем качества. Он также позволяет передавать-принимать сигнал по двум каналам, зато может обходиться совсем без АЦП, поскольку децимация (выборка) может происходить непосредственно в микрофоне. А вот для демодуляции PDM сигнала необходим процессор цифрового сигнала (digital signal processor – DSP).

Датчики давления MEMS

Тензодатчики давления находят применение в широком диапазоне IoT: от контроля за инфраструктурой умных городов до промышленного производства. Обычно они используются для измерения давления жидкостей и газов. Основой датчика является пьезоэлектрический элемент, к которому организован физический доступ через диафрагму (отверстие) в подложке над или под элементом. Подложка делается гибкой, что позволяет пьезоэлементу под воздействием давления изгибаться, изменяя свое электрическое сопротивление (рис. 3.11).

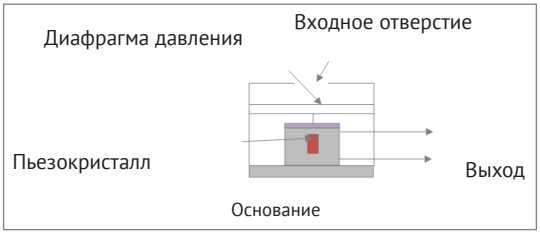


Рис. 3.11 ❖ Анатомия датчика давления

Принцип действия этого типа датчиков аналогичен другим, перечисленным в этой главе. Ток от внешнего источника питания, прошедший через датчик, измеряется посредством моста Уитстона. Такие мосты могут могут быть выполнены в двух-, четырех- или шестипроводных модификациях. Мост измеряет изменения потенциала в цепи, когда пьезоэлектрическая подложка изгибается и меняет свое сопротивление (рис. 3.12).

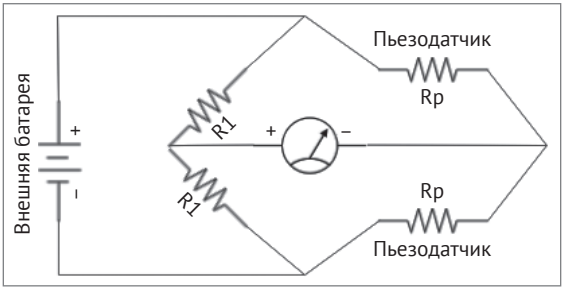


Рис. 3.12 ❖ Мост Уитстона, используемый для измерения сопротивления датчика давления MEMS

Интеллектуальные оконечные точки IoT

До сих пор мы рассматривали очень простые датчики, которые просто возвращали информацию в двоичной или аналоговой форме, которую еще надо обрабатывать. Однако существуют устройства и датчики IoT, которые обладают значительной вычислительной мощностью и производительностью, достаточной для самостоятельной обработки данных и принятия решений. Интеллектуальные датчики имеют в своем составе такие устройства, как видеокамеры и даже целые системы видеонаблюдения. Такие датчики могут производить значительные объемы вычислений благодаря встроенным процессорам цифрового сигнала: ПЛИС (программируемая логическая интегральная схема) и пользовательским ASIC (application-specific integrated circuit – интегральная схема специального назначения). В этом разделе мы детально рассмотрим одну из форм интеллектуального сенсора: видеосистему.

Видеосистема

В отличие от простых датчиков, рассмотренных ранее, видеосистемы намного сложнее, поскольку требуют серьезного аппаратного обеспечения, оптики и светочувствительных матриц изображения. Такие системы начинаются с объектива, посредством которого осуществляется наблюдение. Объектив обеспечивает не только резкость изображения, но и большую светочувствительность активного элемента. В современных системах видения используется один из двух типов чувствительных элементов: **приборы с зарядовой связью** (ПЗС) или **комплементарные металлооксидные полупроводники** (КМОП). Разницу между КМОП и ПЗС можно выразить как:

- **ПЗС** (CCD): сигнал от датчика к периферийному оборудованию микросхемы передается с помощью аналого-цифрового преобразователя. Эти датчики создают изображения с высоким разрешением и малым шумом. Зато они потребляют значительную мощность (100x от КМОП) и сложны в изготовлении;
- **КМОП** (CMOS): изображение строится из отдельных пикселей (точек), каждый пиксель формируется отдельным транзистором, то есть каждый пиксель считывается отдельно. КМОП более восприимчив к шуму, но очень энергетически экономичен.

Большинство видеодатчиков, представленных на современном рынке, построено по КМОП-технологии. Такой датчик встроен в кремниевую подложку и выглядит как двумерная матрица транзисторов, расположенных рядами и столбцами. Каждая ячейка такой матрицы состоит из трех фотодиодов для трех цветов – красного, зеленого и синего. Каждый фотодиод снабжен микролинзой, которая фокусирует случайные лучи определенного цвета, ослабляя другие. Эти линзы далеко не идеальны, в них возникает хроматическая аберрация, то есть разные длины волн преломляются с разной скоростью, что приводит к размытию изображения. Линзы могут также вызывать искажения изображения вследствие подушкообразных искажений.

Далее мы рассмотрим действия, предпринимаемые для фильтрации шумов, нормализации и оцифровки изображения, чтобы придать ему форму удобную для использования. Основа процесса – это процессор видеосигналов изображения (image signal processor – ISP). Вся процедура может выполняться в следующем порядке (см. рис. 3.13).

Обратите внимание на многочисленные преобразования на каждом этапе формирования изображения: эти преобразования производятся для каждого пикселя. Для обработки такого объема данных требуются значительные вычислительные мощности. Ниже перечислены функциональные блоки, необходимые для обработки изображения:

- **аналого-цифровое преобразование**: усиление сигнала датчика с последующим его преобразованием в цифровую форму (10 бит). Данные, представляющие захваченное изображение, считываются из матрицы фотодиодов, как из двумерной таблицы, состоящей из строк и столбцов;

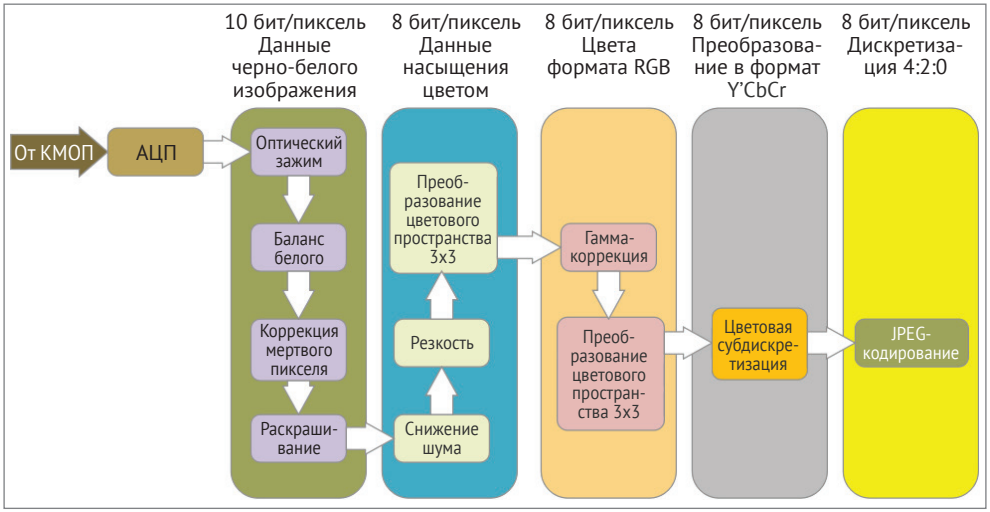


Рис. 3.13 ❖ Датчик изображения: типичный процессор видеосигналов изображения для создания цветного видео

- **оптический зажим**: удаляет эффекты затемнения, возникающие вследствие засветки отдельных пикселей датчика (sensor black level);
- **баланс белого**: имитирует хроматическое восприятие человеческим глазом различных цветовых температур, в результате чего нейтральные тона и выглядят нейтральными. Выполняется с использованием преобразования матриц;
- **коррекция мертвого пикселя**: определяет пиксели, вышедшие из строя, и компенсирует их потерю с использованием интерполяции, значения мертвых пикселей заменяются усредненными значениями соседних;
- **раскрашивание (Debayer filtering) и сбор мозаичного изображения (demosaicing)**: придает монохромному изображению цвета, причем насыщенность зеленого цвета определяется как функция от уровней красного и синего цветов. Создает плоское изображение посредством чересстрочной развертки. Для получения более резких изображений с четко очерченными границами и контурами используются более сложные алгоритмы;
- **снижение шума**: все датчики создают шум. Шум может быть связан с неравномерностью чувствительности пикселей на уровне транзистора или утечкой фотодиода. Это явление образует темные области. Существуют и другие формы шума. На данном этапе удаляются белый и когерентный шумы путем введения промежуточного фильтра (массив 3x3) для всех пикселей. В качестве альтернативы иногда используют фильтр пятен (despeckle filter), требующий сортировки пикселей, но существуют и другие методы. Все перечисленные методы обрабатывают все пиксели по всей матрице в целом;

- **резкость**: производится размытие изображения с использованием матричного умножения, а затем в сочетании с детализацией по отдельным областям создается эффект наведения резкости;
- **преобразование цветового пространства 3×3**: преобразование цветового пространства в данные, соответствующие стандарту формата RGB;
- **гамма-коррекция**: исправляет нелинейный отклик датчика изображения КМОП на данные RGB для различной освещенности. Гамма-коррекция использует справочную таблицу (lookup table – LUT) для интерполяции и исправления изображения;
- **преобразование цветового пространства 3×3**: дополнительное преобразование цветового пространства из формата RGB в формат Y'CbCr. YCC был выбран, поскольку Y можно хранить с более высоким разрешением, чем CbCr, без потери визуального качества. Структура дискретизации 4:2:2;
- **цветовая субдискретизация (Chroma subsampling)**: устраняет нелинейность RGB-тонов, корректирует изображения для имитации изображения на других носителях, таких как пленка. Таким образом улучшается соответствие тонов и качества;
- **JPEG-кодирование**: стандартный алгоритм сжатия JPEG.

Здесь следует подчеркнуть, что это хороший пример того, насколько сложным может быть датчик, какой объем данных, оборудования и сложности можно соотнести с простой видеосистемой. Объем данных, обрабатываемых видеосистемой с частотой 60 кадров в секунду при разрешении 1080p, просто огромен. Предполагая, что все этапы обработки (кроме сжатия JPEG) производятся посредством ISP, а так как это одна микросхема (ASIC) и все этапы выполняются за один цикл, то общий объем обрабатываемых данных составит 1,368 ГБ/с. А с учетом сжатия JPEG, которое производится пользовательским CPU/DSP (центральный процессор/сопроцессор), объем данных составит уже более 2 ГБ/с. Никто и никогда не передает необработанный видеопоток (raw Bayer) в облако для обработки – эта работа должна выполняться как можно ближе к видеодатчику.

Слияние датчиков

Ко всем сенсорным устройствам, описанным в этой главе, применима концепция **слияния датчиков**. Это процесс объединения нескольких различных датчиков в целях получения большего объема информации, чем может обеспечить один датчик. В пространстве IoT это важно, поскольку, например, единичный тепловой датчик понятия не имеет о том, что именно вызывает быстрое изменение температуры. Но в сочетании с другими датчиками, например, датчиками PIR, фиксирующими движение и интенсивность освещенности, система IoT может понять, что в определенной области собралось большое количество людей и ярко светит солнце, на этом основании она может принять решение об

усилении циркуляции воздуха. А один термодатчик просто зафиксирует текущее значение температуры без какого-либо осознания того, что температура растет из-за того, что собрались люди и светит солнце.

На основе большего количества данных от большего количества датчиков, соответственно коррелированных во времени, система может принимать более взвешенные решения. Это одна из причин того, что количество датчиков, помещенных в облако IoT, растет, вызывая рост объемов данных. Датчики становятся дешевле, легче интегрируются, и на примере TISensorTag легко видеть, как комбинация датчиков облегчает общее видение.

Существует два режима слияния датчиков:

- **централизованный**: данные передаются в центральный офис, где и происходит их слияние (пример – облачные технологии);
- **децентрализованный**: корреляция данных непосредственно в датчике (или рядом с ним).

В основе корреляции данных датчика лежит центральная предельная теорема, на основании которой два независимых измерения x_1 и x_2 объединяются с учетом их дисперсий (отклонений от нормы), чтобы получить третье значение x_3. То есть, это просто расчет средневзвешенного значения первых двух величин:

$$x3 = (\sigma_1^{-2} + \sigma_2^{-2})^{-1}(\sigma_1^{-2}x_1 + \sigma_2^{-2}x_2).$$

Другими методами слияния датчиков являются фильтры Калмана и Байесовские сети.

Устройства ввода

Существует еще множество типов датчиков, рассмотрения которых мы не касаемся в этой главе. Это различные газовые анализаторы, датчики влажности, датчики радиоактивного излучения, датчики дыма, ультразвуковые датчики и т. д. Тем не менее, эта глава должна дать читателю довольно основательные представления о сенсорных устройствах и снабдить его знаниями, необходимыми при их выборе.

До этого момента мы обсуждали оконечные точки, представленные датчиками. Эти устройства отправляют постоянный поток данных на центральное устройство или в облако. IoT строится на основе двунаправленных систем. Входные данные могут передаваться в оконечную точку из облака, или, наоборот, данные могут быть отправлены оконечной точкой другим абонентам облака. В этом коротеньком разделе мы рассмотрим основные приводы и выводные устройства.

Устройства вывода

Выходные устройства в экосистеме IoT могут быть практически любыми: от простого светодиода до полноценной видеосистемы. К другим типам выход-

ных сигналов относятся исполнительные механизмы, шаговые двигатели, громкоговорители и аудиосистемы, промышленные клапаны и т. д. Разумеется, эти устройства нуждаются в различных системах управления различной сложности. В зависимости от типа выхода и используемого варианта использования, также следует ожидать, что большая часть контроля и управления должна производиться непосредственно вблизи устройства (в противоположность полному контролю в облаке). Например, видеосистема может передавать данные облачных провайдеров, но для этого требуется оборудование вывода и буферизации.

В общем случае системам вывода требуются значительные объемы электроэнергии для ее преобразования в механическое движение, тепловую энергию или в свет. Например, небольшой соленоид для управления потоком жидкости или газа при напряжении питания 9–24 В будет потреблять ток примерно в 100 мА, при этом создаст направляющее усилие всего в пять ньютонов. А промышленные соленоиды работают от напряжения в сотни вольт.

Функциональные примеры (все вместе)

Набор датчиков довольно бесполезен, если данные, которые они собирают, не могут передаваться и обрабатываться. Независимо от того, установлен ли локальный или встроенный контроллер или данные отправляются на более высокий уровень, для построения полноценной системы требуется больший набор оборудования, чем просто датчики и котроллер.

Наиболее распространенные интерфейсы ввода-вывода, используемые в датчиках, это I^2C, SPI, UART, хотя существуют и другие. Для таких устройств, как видеосистемы с высоким разрешением и большой кадровой частотой, необходимы высокоскоростные IO-интерфейсы, такие как MIPI, USB или даже PCI-Express. Датчики могут также использоваться с оборудованием беспроводной связи, таким как Bluetooth, Zigbee или 802.11. Все это требует дополнительных компонентов, которые мы и рассмотрим в этом разделе.

Функциональный пример – TI SensorTag CC2650

Texas Instruments CC2650 SensorTag является хорошим примером сенсорного модуля IoT для разработки и проектирования. Ниже приведено описание его функционала и встроенных датчиков:

- входной датчик:
  - датчик освещенности (TI Light Sensor OPT3001);
  - инфракрасный датчик температуры (TI Thermopile infrared TMP007);
  - датчик температуры окружающей среды (TI light sensor OPT3001);
  - акселерометр (Invensense MPU-9250);
  - гироскоп (Invensense MPU-9250);
  - магнитометр (Bosch Sensortec BMP280);
  - альтиметр (датчик высоты)/Датчик давления (Bosch SensortecBMP280);

  - датчик влажности (TI HDC1000);
  - микрофон MEMS (Knowles SPH0641LU4H);
  - магнитный датчик (Bosch SensorTec BMP280);
  - 2 интерфейса GPIO;
  - герконовое реле (Meder MK24);
- устройства вывода:
  - зуммер/динамик;
  - 2 светодиода;
- коммуникации:
  - Bluetooth Low Energy (Bluetooth Smart);
  - сетевой протокол – Zigbee;
  - протокол взаимодействия с беспроводными сетями – 6LoWPAN.

Весь комплект рассчитан на питание от одной батареи-монеты CR2032. Устройство может работать в режиме маяка (iBeacon) или использоваться в качестве оповещателя. На рис. 3.14 приведена блок-схема модуля CC2650 SensorTag.

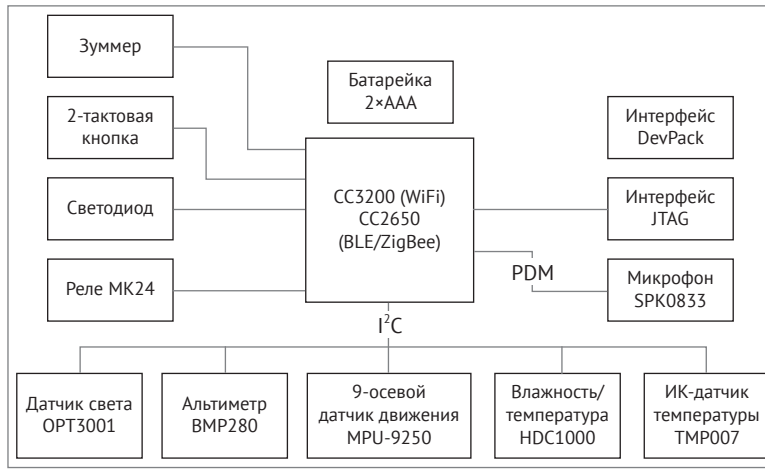


Рис. 3.14 ❖ TI CC2650 SensorTag.
Производитель Texas Instruments, TI Multi-Standard CC2650 SensorTag Design Guide. Texas Instruments Incorporated, 2015

На рис. 3.15 приведена блок-схема устройства для реализации аудио- и видеоконференций (Multipoint Control Unit – MCU). MCU обеспечивает возможность ввода-вывода, сигнал обрабатывается с процессором ARM Cortex M4, для подключения датчиков имеет различные интерфейсы.

Устройство оснащено несколькими датчиками, системами связи, интерфейсами, но вычислительная мощность невелика. В нем используется модуль обработки TI (MCU CC265), который включает в себя небольшой процессор ARM Cortex M3 с флеш-памятью объемом 128 КБ и 20 КБ SRAM. Обладает чрезвы-

чайно низким энергопотреблением. Но, несмотря на высокую энергоэффективность, объем обрабатываемой информации невелик в силу ограниченности ресурсов. Как правило, такие устройства сопровождаются шлюзами, маршрутизаторами, смартфонами или другими интеллектуальными устройствами. Сенсорные устройства бюджетного исполнения ориентированные на низкое энергопотребление и малую стоимость, не имеют ресурсов для поддержания более требовательных приложений, таких как стеки протоколов MQTT, агрегация данных, сотовая связь или аналитика. Подобные устройства предназначены для слежения за полевыми датчиками, они разрабатывались исключительно в целях экономии средств и снижения затрат.

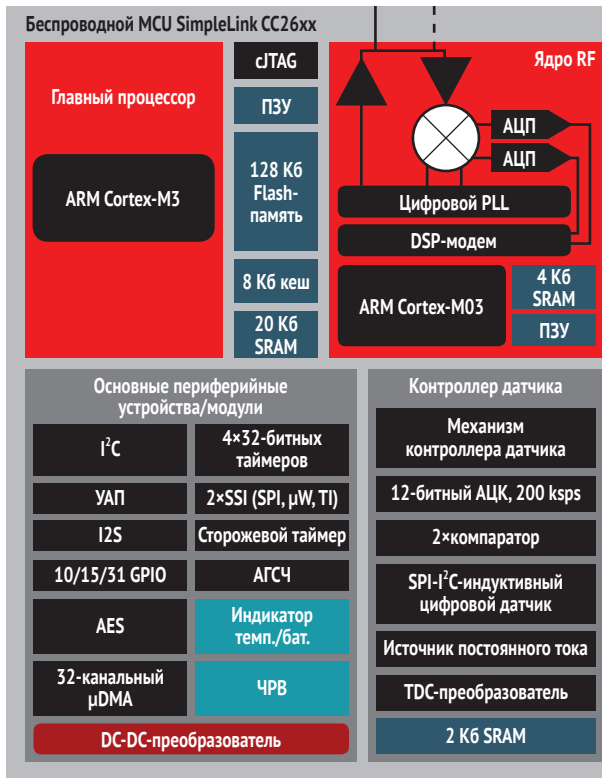


Рис. 3.15 ❖ TI CC2650. Блок-схема MCU.
Предоставлено Texas Instruments, TI Multi-Standard CC2650 SensorTag Design Guide. Texas Instruments Incorporated, 2015

Между датчиком и контроллером

Во многих предыдущих примерах сигнал датчика перед отправкой на следующий уровень требует усиления, фильтрации и калибровки. Кроме этого, как

правило, требуется аналого-цифровой преобразователь. На рис. 3.16 приведен простой 24-разрядный АЦП с опорным напряжением 5 В.

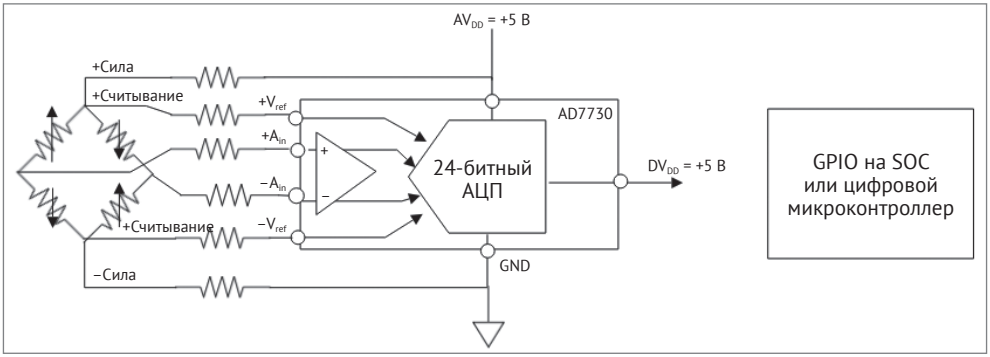


Рис. 3.16 ❖ Мост Уитстона: подключен к аналого-цифровому преобразователю AD7730, который выступает в качестве входного устройства для подключения к микроконтроллеру или другому входному преобразователю

Далее с выхода АЦП сигнал может поступать на вход устройства, которое преобразует его в импульсно-модулированный сигнал или команды какого-нибудь последовательного интерфейса, например, I^2C, SPI или UART, которые уже и будут переданы непосредственно микроконтроллеру или процессору цифровых сигналов. Хорошим примером всего вышесказанного может служить инфракрасный термочувствительный датчик Texas Instruments (TMP007). Это бесконтактный MEMS-датчик температуры, который поглощает инфракрасное излучение и преобразует его в напряжение. Он предназначен для точного определения температуры окружающей среды в пределах от –40 до +125 °C. Все компоненты, описанные в этом разделе, представлены на рис. 3.17.

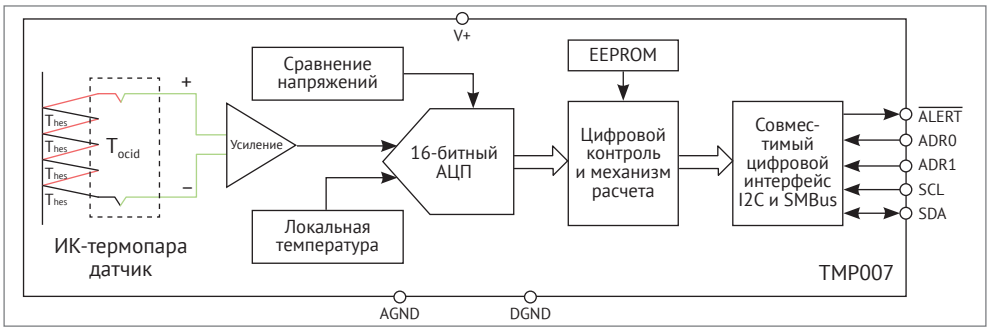


Рис. 3.17 ❖ TI TMP007. Предоставлено Courtesy от Texas Instruments. TI Multi-Standard CC2650 Руководство по проектированию датчиков. Texas Instruments Incorporated, 2015

Источники энергии и управление питанием

Использование мощных датчиков и периферийных устройств связано с одной проблемой. Когда мы говорим о миллиардах элементов IoT, понятно, что некоторые из них будут установлены в очень отдаленных районах, и тогда их питание становится очень проблематичным.

Кроме того, при развертывании IoT некоторые датчики могут располагаться под водой или встраиваться в существующие инфраструктуры, что еще больше усложняет вопросы питания. В этом разделе мы рассмотрим концепции производства электроэнергии и управления питанием. Обе эти концепции чрезвычайно важны в плане IoT.

Управление питанием

Управление питанием – очень широкая тема, которая касается как программного, так и аппаратного обеспечения. При развертывании IoT очень важно понимать роль эффективного управления энергопотреблением удаленных устройств и устройств долговременной эксплуатации и владеть приемами управления им.

Проектировщик должен строить бюджет энергопотребления, учитывая следующие факторы:

- активная мощность датчика;
- частота сбора данных;
- потребляемая мощность беспроводной радиосвязи, необходимая для обеспечения покрытия заданной площади;
- частота связи;
- мощность микропроцессора или микроконтроллера в зависимости от тактовой частоты ядра;
- мощность пассивной составляющей;
- потери энергии от утечек или неэффективности питания;
- резервирование электроэнергии для питания приводов и двигателей.

Бюджет просто отражает суммарное энергопотребление этих потребителей, обеспечиваемое источником питания (батареи). С течением времени режим работы батареи не линеен. Поскольку батарея теряет энергоемкость при разрядке, ее напряжение падает криволинейно. Это создает проблемы для систем беспроводной связи. Если аккумулятор разрядится до некоторого критически минимального напряжения, радио или микропроцессор, не имея порогового напряжения питания, просто отключится.

Например, TI SensorTag C2650 имеет следующие характеристики мощности:

- режим ожидания: 0,24 мА;
- работа со всеми отключенными датчиками: 0,33 мА;
- светодиоды;
- все датчики со скоростью 100 мс/измерение и широковещательные устройства BLE: 5,5 мА:

- датчик температуры: 0,84 мА;
- датчик освещенности: 0,56 мА;
- акселерометр и гироскопы: 4,68 мА;
- барометрический датчик: 0,5 мА.

TI SensorTag использует стандартную аккумуляторную батарею CR2032, рассчитанную на 240 мА/ч. Поэтому ожидается, что ее максимальный срок службы составит около 44 часов. Но скорость снижения энергоемкости (и, соответственно, напряжения), для батарейных устройств, величина нелинейная, в этом мы убедимся при рассмотрении емкости Пейкерта.

Методы управления питанием широки и разнообразны. Это и использование неэлектронных реле времени, снижение тактовой частоты процессоров или микроконтроллеров, регулирование частоты, ограничение ширины полосы радиовещания, стратегии отсрочки сеансов связи и различные режимов сна. Эти методы широко используются, а в вычислительной технике уже стали общепринятой практикой.

Вышеописанные методы уже стали классикой. Они минимизируют энергопотребление с помощью динамического управления напряжением, частотного регулирования и других схем. Но существуют и новые методы, с которыми следует познакомиться: это приблизительный расчет и вероятностное проектирование. Обе эти схемы основаны на том факте, что абсолютная точность в показаниях датчика не всегда необходима, особенно если сигнал подлежит обработке для передачи посредством беспроводной связи. Округление вычислений может выполняться на аппаратном или программном уровне простым округлением до целого. Это тем более приемлемо, когда речь идет об адресах или о произведении чисел (например, значение 17,962 довольно близко к 17,970). Вероятностное проектирование предполагает лишь заданную степень надежности, а не максимально возможную, что снимает многие конструктивные ограничения. Оба метода в совокупности способны снизить энергопотребление почти экспоненциально в сравнении с обычными схемами.

Воспроизводство электроэнергии

Воспроизводство электроэнергии, концепция не новая, но в плане IoT очень важная. По сути, любая система, следящая за изменениями условий окружающей среды (например, от жары к холоду, получение радиосигналов, освещенность), может преобразовывать наблюдаемые формы энергии в электрическую. В некоторых устройствах это используется как единственный источник энергии, а есть и гибридные системы, которые используют такое преобразование для подзарядки батарей и продления их срока службы. С другой стороны, произведенная таким образом энергия может сохраняться и использоваться экономно для питания низкоэнергетических устройств, например датчиков в IoT. В любом случае, системы должны быть эффективны при преобразовании и сохранении энергии. Это требует расширенных возможностей в управлении питанием. Например, если система производства электроэнергии использует технологию пьезоэлектрической подложки, встроенной в тротуар, то должна быть предусмотрена компенсация на случай, когда пешеходное движение будет недостаточным. Постоянная связь с преобразователями энергии тоже ве-

дет к увеличению энергопотребления, которое также надо компенсировать. При развертывании IoT, как правило, используются передовые технологии управления питанием, исключающие полную потерю функциональности систем. Часто используются такие методы, как ожидание в режиме пониженного энергопотребления, сокращение потерь, периодические включения и выключения с использованием реле времени. На рис. 3.18 ниже показана область энергопотребления, которая идеально подходит для питания от преобразователей, а также технологии, которые могут это обеспечить. Проектировщик должен позаботиться о том, чтобы система не испытывала недостатка в электроэнергии, но и не имела ее в избытке.

Системы преобразования и воспроизводства электроэнергии, в целом, имеют низкий потенциал и эффективность. Поэтому вопрос о воспроизводстве электроэнергии целесообразно рассматривать лишь в условиях действительного избытка неиспользуемой, бросовой энергии, например, в условиях промышленного производства.

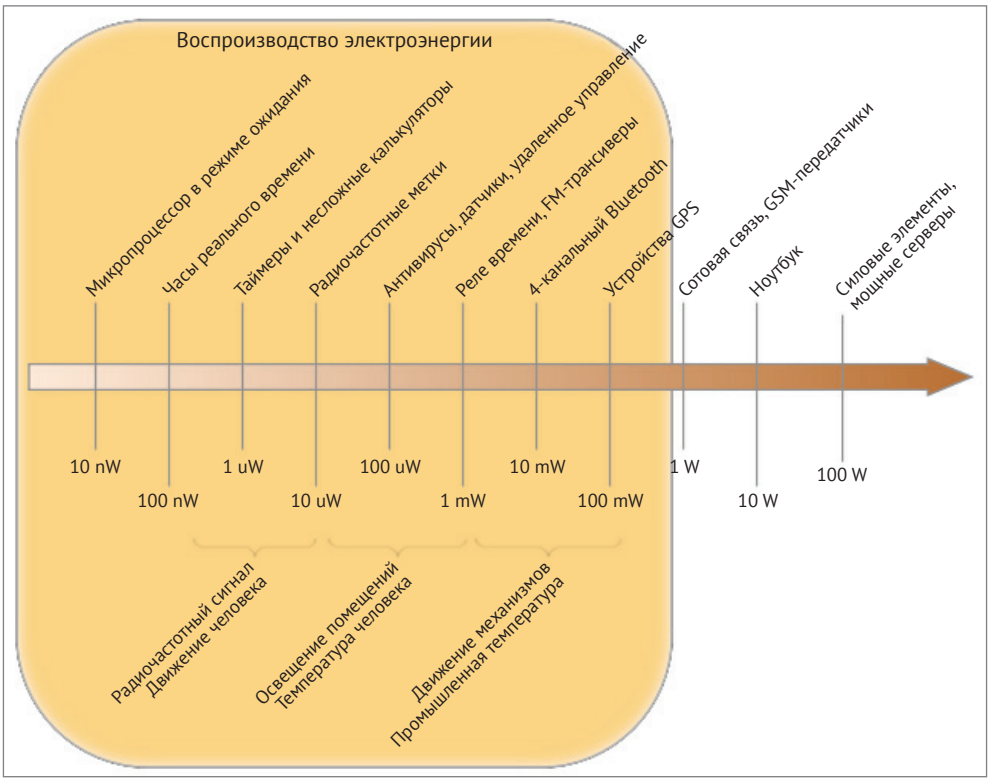


Рис. 3.18 ❖ Область энергопотребления, подходящая для питания воспроизведенной электроэнергией. На рисунке показано типичное потребление энергии для различных устройств

Преобразование солнечной энергии

Энергия света, естественного или искусственного, легко может быть использована в качестве источника электроэнергии. Ранее в этой главе мы уже обсуждали фотодиоды, когда рассматривали светочувствительные датчики. Такие диоды, но в больших количествах, можно использовать для создания традиционных солнечных батарей. Их способность генерировать энергию напрямую зависит от их площади. Наиболее эффективны батареи, установленные под прямыми солнечными лучами, а не в условиях искусственного освещения. Классифицируются такие панели по максимально возможной мощности на входе, оцениваемой в Вт.

Эффективность преобразования солнечной энергии зависит от солнечной активности, которая меняется сезонно и географически. Например, на Юго-Западе США фотоэлектричество составляет значительную долю общего электропотребления. Национальной лабораторией по возобновляемым источникам энергии при Министерстве энергетики США (**www.nrel.gov**) была выпущена карта Соединенных Штатов с указанием объемов фотоэлектрических ресурсов. Эта карта показана на рис. 3.19.

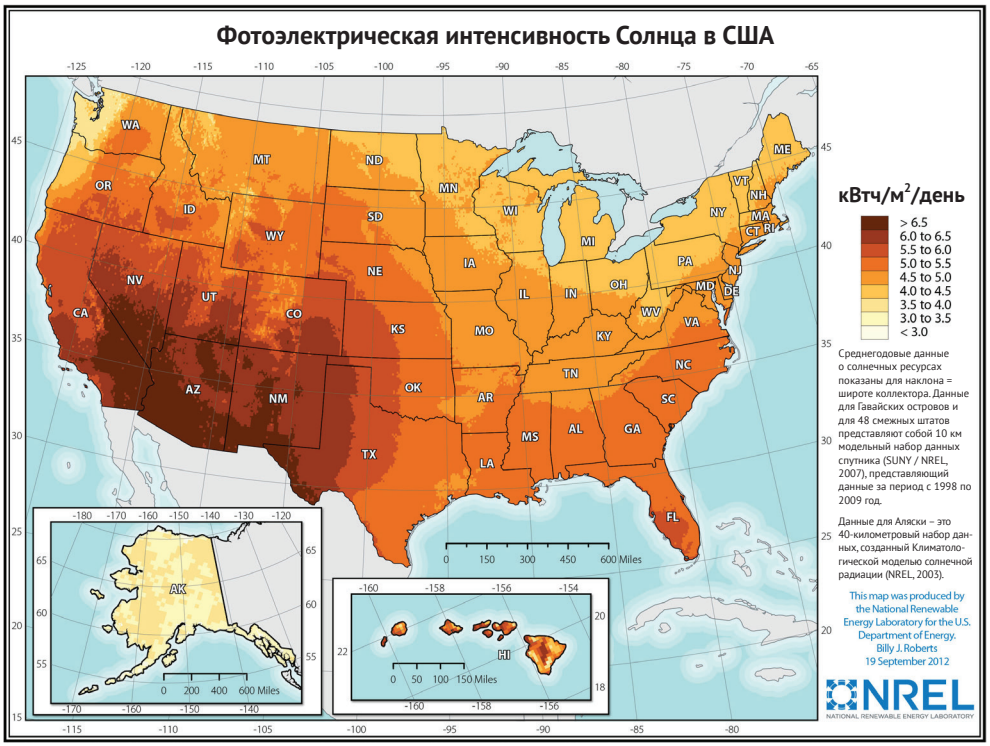


Рис. 3.19 ❖ Карта Соединенных Штатов с указанием плотности солнечной энергии кВт·ч/м^2 1998–2009 гг.

В юго-западных регионах Соединенных Штатов солнечная активность высока, обычно нет облачных барьеров и преобладают хорошие атмосферные условия. А вот на Аляске плотность солнечной энергии мала. Солнечные фотоэлектрические элементы обычно неэффективны. Обычно их КПД колеблется в пределах от 8% до 20%, при этом показатель в 12%, является наиболее типичным. Тем не менее, солнечная батарея площадью 25 см^2 при максимально возможной освещенности может генерировать мощность до 300 мВт. Другим фактором, влияющим на генерируемую мощность, является угол падения солнечных лучей. Максимальной эффективности солнечный коллектор достигает при перпендикулярном расположении относительно солнечного света. При изменении угла падения, по мере движения Солнца, эффективность батареи падает. Например, коллектор с максимальным КПД в 12% при наклоне лучей в 60° будет работать с КПД в 9,6%.

Основа солнечного коллектора – это солнечный элемент, который представляет собой простой полупроводник с *p*-*n*-переходом и полностью аналогичен ранее описанным фотоэлектрическим датчикам. Как объяснялось выше, в таком элементе при облучении светом между областями *p* и *n* генерируется электрический потенциал.

Пьезомеханическое преобразование

Как уже упоминалось ранее, пьезоэлектрические эффекты могут использоваться в качестве датчиков, но они же могут быть использованы и для генерации энергии. Механические деформации, вызываемые движением, вибрацией и даже звуком могут быть преобразованы в электроэнергию. Такие генераторы могут использоваться в умных дорогах, даже если они встроены в бетон. Подобные устройства генерируют мощность порядка милливатт и, следовательно, подходят для очень небольших систем, предназначенных для сбора и хранения энергии. Весь процесс может быть исполнен с применением пьезомеханических устройств MEMS, электростатических и электромагнитных систем.

Принцип действия электромагнитной системы основан на законе Фарадея, гласящем, что изменение магнитного потока через катушку с проводом индуцирует в ней электрический ток. Чтобы обеспечить такие изменения, вибрация передается либо на катушку, либо на магнит. К сожалению, такая схема обеспечивает слишком малое напряжение.

Электростатические системы используют эффект, возникающий при изменении расстояния между двумя емкостными пластинами, на которых поддерживается постоянное напряжение или заряд. Любая вибрация пластин вызывает изменение расстояния между ними, возникающая при этом энергия (E) может быть проиллюстрирована посредством следующей модели:

$$E = \frac{1}{2}QV^2 = \frac{Q^2}{2C}.$$

Здесь Q – постоянный заряд на пластинах, V – постоянное напряжение, C – электрическая емкость. Емкость может быть выражена через длину пластины

L_w, относительную диэлектрическую проницаемость ε_0 и расстояние между пластинами d следующим соотношением:

$$C = \varepsilon_0 L_w / d.$$

Преимущество электростатического преобразования заключается в том, что оно может быть масштабируемым и экономически целесообразным в условиях микромашиностроения и производства полупроводников.

Последний метод механико-электрического преобразования – это пьезомеханический метод, о котором уже упоминалось. Эта же базовая концепция, что используется в датчиках, применяется и при воспроизводстве электроэнергии. Пьезомеханическое устройство MEMS, противодействуя приложенному к нему усилию, генерирует электрическую энергию.

Еще одним аспектом преобразования механической энергии в электрическую является необходимость выпрямления и сглаживания полученного напряжения. Обычно для этих целей используется пассивный выпрямитель с подключенным конденсатором большой емкости в качестве сглаживающего фильтра. Другие формы воспроизводства электроэнергии не нуждаются в таких преобразованиях.

Преобразование радиочастот

Беспроводное энергоснабжение с помощью **радиочастот** (RF) ведется уже в течение многих лет, тому пример радиометки (RFID tags). RFID, как правило, находится в непосредственной близости от приемопередатчика, который не только обменивается с ней информацией, но и снабжает ее энергией.

Но удаленным устройствам нужно воспроизводить электроэнергию из широковещательных трансляций. Передача в широковещательном диапазоне ведется почти повсеместно: это и телевидение и сигналы сотовой связи, и радио. Воспроизводство электроэнергии из радиочастот – это наиболее трудная задача в сравнении с другими формами генерации, поскольку радиочастотные сигналы имеют наименьшую плотность энергии. Прием радиочастотных сигналов производится посредством антенны, настроенной на определенную полосу частот. Типичный диапазон частот, использующийся для радиовещания, составляет от 531 до 1611 кГц (диапазон радиочастот AM).

Преобразование тепла

В электроэнергию может быть преобразована тепловая энергия любого устройства, которому требуется отвод теплоты. Обычно для этого используются два основных процесса:

- термоэлектрический: прямое преобразование тепловой энергии в электрическую энергию вследствие эффекта Зеебека;
- термический: также известен как термотуннелирование. Электроны выбрасываются (эмитируются) из нагреваемого электрода и в направлении холодного.

Термоэлектрический эффект (эффект Зеебека) возникает в проводящих материалах при наличии градиента температур. Поток носителей из горячей в холодную область между двумя несходными электрическими проводниками создает разность потенциалов. Термопара, или **термоэлектрический генератор** (ТЭГ), может эффективно генерировать напряжение даже вследствие разности температуры человеческого тела и температуры окружающей среды. Разность температур в 5 °C может генерировать мощность в 40 мкВт при напряжении 3 В. Когда тепло передается от горячего электрода к холодному, горячая сторона индуцирует поток электронов в направлении холодного электрода. В современных термоэлектрических устройствах, в основном, используется теллурид висмута, *n*- и *p*-типа. В таких устройствах один электрод элемента (тоже называемого термопарой) подвергается воздействию источника тепла, а другой изолирован. Энергия, производимая термопарой, пропорциональна квадрату напряжения и разнице температур между электродами. Математически это можно выразить следующим уравнением:

$$V = \int_{T_L}^{T_H} [S_1(T) - S_2(T)]dT.$$

Здесь S_1 и S_2 – коэффициенты Зеебека для каждого из двух материалов (*n*- и *p*-типа), из которых построена термопара; $T_H - T_L$ – разность температур. Поскольку коэффициент Зеебека – это функция температуры и существует разность температур, то результатом является разность напряжений. Это напряжение, как правило, очень мало, поэтому для образования термоэлемента последовательно соединяется несколько термопар.

Одной из существенных проблем современных термопар является их низкая эффективность преобразования энергии (менее 10%). Однако и их преимущества очевидны: это и небольшие размеры, и простота изготовления, и невысокая стоимость. Продолжительность срока их службы – более 100 000 часов. Основная же проблема заключается в нахождении относительно постоянного источника тепловой дисперсии (разности температур). Использование такого устройства в течение нескольких сезонов с разными температурами является сложной задачей. Генерируемая мощность таких устройств IoT обычно лежит в диапазоне 50 мВт.

Термическая генерация основана на эффекте выброса (эммитации) электронов горячим электродом в сторону холодного вследствие увеличения энергии электронов до уровня достаточного для преодоления потенциального барьера. Потенциальный барьер – это физическая характеристика материала. Этот метод лучше всего использовать, когда имеется источник тепловой энергии значительной мощности. Хотя его эффективность выше эффективности термоэлектрических систем, энергия, необходимая для преодоления потенциального барьера, делает его в целом непригодным для датчиков IoT. Можно рассмотреть альтернативные схемы, такие как квантовое туннелирование, но в настоящее время они все еще остаются предметом для исследований.

Хранилище энергии

Типичным хранилищем энергии для датчика IoT служат батарея или суперконденсатор (ионистор). При рассмотрении архитектуры мощного датчика необходимо учитывать несколько аспектов:

- энергия, необходимая для питания силовой подсистемы: сумеет ли батарея ее обеспечить;
- энергоемкость аккумулятора;
- доступность: если устройства замурованы в бетон, будет ли целесообразна дорогостоящая замена, если они выйдут из строя, тем более, что их возможности регенерации энергии весьма скромны;
- вес: этот параметр весьма критичен, если устройство предназначено, например, для беспилотного летательного аппарата;
- как часто будет заряжаться аккумулятор;
- является ли возобновляемая форма энергии постоянно доступной или эпизодической, как, например, солнечная;
- характеристики мощности батареи: как энергия батареи будет меняться со временем, в процессе разрядки;
- установлен ли датчик в термически ограниченной среде, что может повлиять на срок службы батареи и ее надежность;
- гарантирует ли батарея минимально допустимый ток.

Моделирование энергии и мощности

Емкость аккумулятора измеряется в ампер-часах. Упрощенное уравнение для оценки срока службы источника питания можно записать так:

$$t = \frac{C_p}{I^n}.$$

Здесь C_p – емкость Пейкерта; I – ток разряда; n – показатель Пейкерта. Эффект Пейкерта, как известно, помогает предсказать срок службы батареи, когда емкость батареи уменьшается с разной скоростью по мере увеличения разряда. Уравнение показывает, что более быстрое разряжение отнимает у аккумулятора больше энергии. И, наоборот, разряжение с низкой скоростью увеличивает эффективное время работы батареи. Приведем пример этого явления. Аккумулятор, рассчитанный производителем на емкость в 100 А·ч, полностью разряжается через 20 часов (при силе тока – 5А). Если его разрядить быстрее (скажем, за 10 часов), его реальная емкость оказалась бы ниже. Но, если разряжать тот же аккумулятор медленнее (например, более 40 часов), его реальная емкость оказалась бы больше. Однако, как показано на графике ниже, эта связь является нелинейной. Показатель Пейкерта обычно лежит между 1,1 и 1,3. Чем больше батарея отличается от идеальной, чем быстрее она разряжается с увеличением тока разрядки, тем выше ее показатель n. Кривая Пейкерта применима к свинцово-кислотной батарее, характеристики которой, для примера, приведены на рис. 3.20.

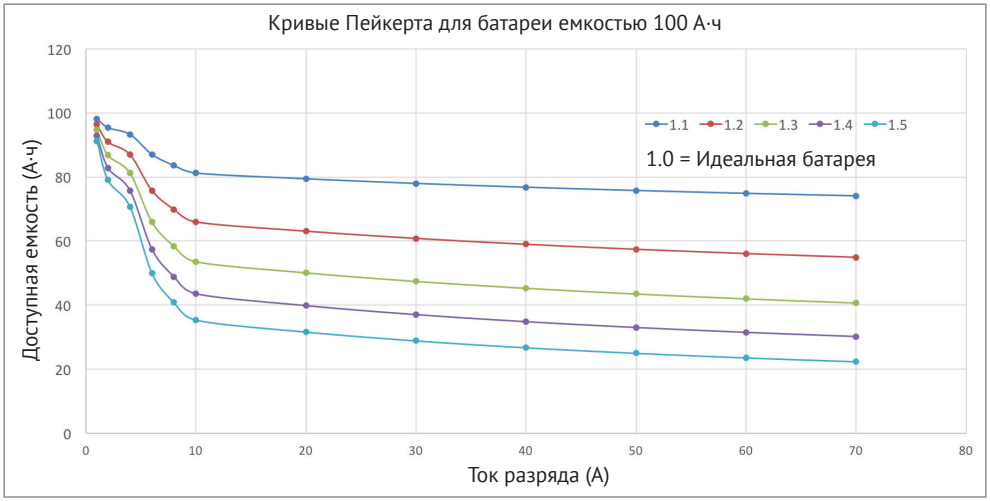


Рис. 3.20 ❖ Кривые Пейкерта с показателем n от 1,1 до 1,5 для батареи, рассчитанной на ток 100 А в течение 20 часов. Кривые показывают снижение мощности по мере увеличения коэффициента Пейкерта

Разница в скорости разряда для различных типов батарей очевидна. Одним из преимуществ щелочных батарей является то, что скорость разряда почти линейна на большой части графика. Литий-ионная батарея имеет ступенчатую функцию производительности, что затрудняет прогнозирование ее разряда. Тем не менее, литий-ионные элементы обеспечивают устойчивый уровень напряжения в течение длительного периода времени и очень удобны для питания электроники в течение всего срока службы (рис. 3.21).

График также иллюстрирует, что кислотно-свинцовая и никель-кадмиевая батареи имеют меньший потенциал, плавное падение мощности, которые можно надежно вычислить. Приведенные кривые носят название емкости Пейкерта.

Температура значительно влияет на срок службы батареи, изменяя характеристики электроактивных носителей. По мере увеличения температуры внутреннее сопротивление батареи уменьшается. Даже когда батареи хранятся, они саморазряжаются, что влияет на общий срок службы батареи.

График Ругони – удобный способ сравнения различных систем хранения энергии по показателям продолжительности обеспечения энергоснабжения и запасаемой мощности, которую система может выдать. На логарифмической шкале по оси ординат откладывается значение плотности энергии (Вт·ч/кг), по оси абсцисс значение плотности мощности (Вт/кг). Это соотношение наглядно демонстрирует различия между устройствами, предназначенными для продолжительного энергообеспечения (батареи) и устройствами, ориентированными на сохранение больших объемов энергии (ионисторы) (рис. 3.22).

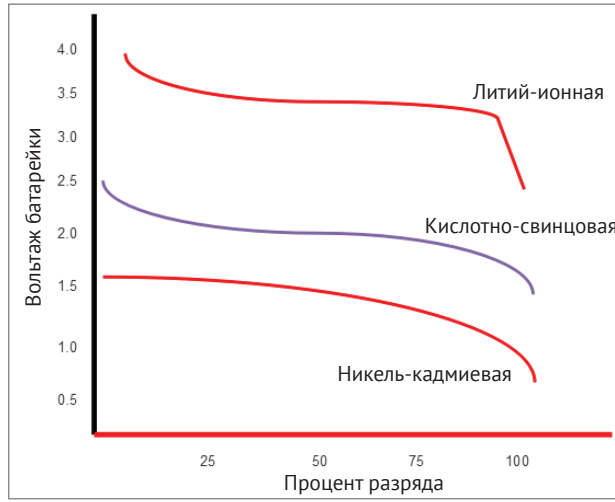


Рис. 3.21 ❖ Пример относительности скорости разряда для различных типов батарей. Литий-ионный элемент обеспечивает почти постоянное напряжение в течение всего срока службы, но имеет резкое падение к моменту исчерпания емкости

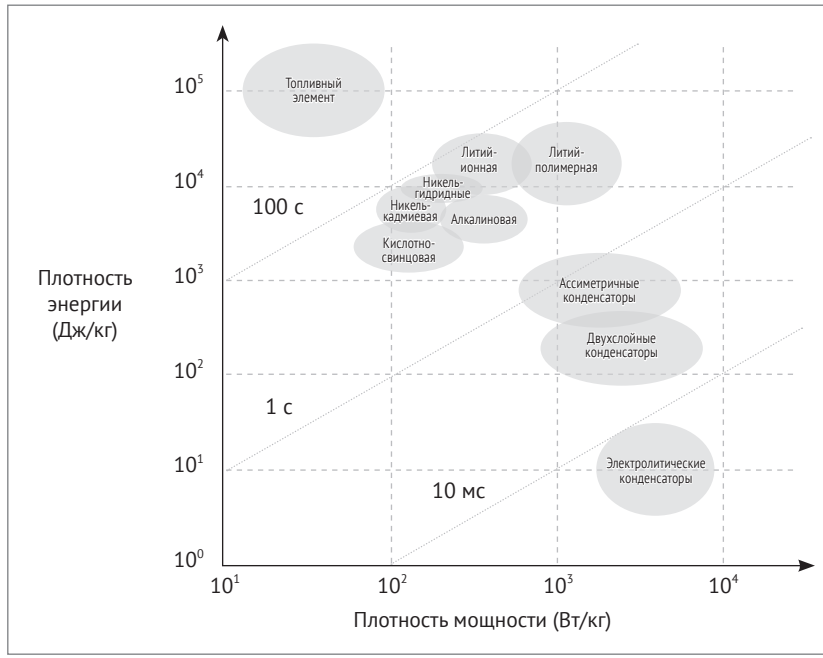


Рис. 3.22 ❖ График Ругони для конденсаторов, ионисторов, аккумуляторов и топливных элементов демонстрирует различия в показателях по возможности длительного энергоснабжения и объемам запасаемой энергии

Литий-ионные батареи имеют более высокую плотность энергии и скорость разряда, чем никель-кадмиевые и никель-гидридные батареи. Конденсаторы обладают очень высокой плотностью мощности, но относительно малой плотностью энергии. Обратите внимание, что график построен на логарифмической шкале и отражает время разряда для различных систем хранения. Изображение предоставлено C. Knight, J. Davidson, S. Behrens «Параметры энергии для узлов беспроводных датчиков», Sensors, 2008, 8 (12), 8037–8066.

Батареи

На сегодняшний день, **литий-ионная** (Li-ion) батарея, благодаря ее плотности энергии, стала стандартным элементом питания в мобильных устройствах. В такой батарее ионы лития физически перемещаются от отрицательного электрода к положительному. Во время перезарядки ионы возвращаются в отрицательную область. Это явление известно как *ионное движение*.

Батареи подвержены старению. Это выражается в потере емкости от начальной после некоторого количества циклов перезарядки (например, потеря 30% емкости после 1000 циклов). Это ухудшение почти напрямую коррелирует с температурой окружающей среды, в высокотемпературной среде потери будут расти. Поэтому проектировщику необходимо позаботиться о соблюдении температурного режима, если он использует литий-ионную батарею.

Другим негативным фактором автономного режима является саморазряд. В процессе эксплуатации в батарее возникают нежелательные химические реакции, которые ведут к потере энергии. Величина и скорость таких потерь зависит от химического состава и температуры. Как правило, срок службы литий-ионного элемента составляет 10 лет (с потерей ~2% в месяц), а щелочной батареи – только 5 лет (15–20% потерь в месяц).

Суперконденсаторы

Суперконденсаторы (или ионисторы) могут сохранять энергию в значительно больших объемах, чем обыкновенные конденсаторы. Обычный конденсатор обладает плотностью энергии около 0,01 Вт·ч/кг. Ионистор имеет плотность энергии от 1 до 10 Вт·ч/кг, по этому параметру он приближается к батареям, плотность энергии которых может достигать 200 Вт·ч/кг. Как и конденсатор, ионистор сохраняет энергию в электростатически заряженных пластинах и, в отличие от аккумулятора, не использует химический перенос энергии. Обычно ионисторы изготавливаются из довольно экзотических материалов, таких, например, как графен, что, естественно, влияет на их стоимость. Очевидное преимущество ионисторов – их скорость зарядки. До полного потенциала они заряжаются за секунды, в то время как, чтобы зарядить литий-ионные батареи на 80%, потребуется несколько минут. Кроме того, ионистор невозможно перезарядить, а вот избыточный заряд литий-ионной батареи может привести к серьезным проблемам безопасности. Ионисторы бывают двух видов:

- **электрические двухслойные конденсаторы** (Electric double-layer capacitors – EDLC): его обкладки изготавливаются из активированного угля, а энергия сохраняется в виде электростатического потенциала;

- **псевдоконденсаторы** (Psuedocapacitors): в качестве диэлектрика между его обкладками используется оксид металла, из которого изготовлены обкладки, для сохранения энергии используется электрохимический перенос заряда.

Еще одно преимущество ионисторов перед батареями – возможность точного прогнозирования времени их службы. Остаточный объем энергии ионистора легко рассчитать по напряжению на его клеммах, оно меняется с течением времени. В отличие от литий-ионной батареи, у которой профиль напряжения сохраняется постоянным, что затрудняет оценку оставшихся объемов энергии. Поскольку профиль напряжения ионистора меняется с течением временем, для компенсации возникающих изменений необходим DC-DC преобразователь.

Основными проблемами ионистора, как и любого другого конденсатора, являются утечка тока и стоимость. Как показано в табл. 3.3 ниже, ионисторы находят довольно широкое применение. Их часто применяют в гибридных решениях, в комплексе с обычными батареями для обеспечения мгновенной мощности (например, при ускорении электромобиля), в то время как аккумулятор обеспечивает транспортную мощность.

Радиоактивные источники энергии

Радиоактивный источник характеризуется высокой плотностью энергии (10^5 кДж/см^3), он генерирует тепловую энергию вследствие высокой кинетической энергии излучаемых частиц. Период полураспада (период времени, по истечении которого мощность источника падает вдвое) такого источника, как, например, цезий-137, составляет 30 лет, а плотность мощности 0,015 Вт/г. Этим методом можно генерировать киловатты мощности, но он неприменим для низкоуровневых мощностей при развертывании IoT. Зато эти технологии успешно применяются в космических аппаратах уже на протяжении десятилетий. Очень перспективно выглядят разработки с использованием пьезоэлектрических MEMS, которые при захвате электронов (источником электронов служит радиоактивный источник) приводят в движение микроарматуру, тем самым производя механическую энергию, которую уже можно преобразовать в электрическую. Вторичным эффектом радиоактивного распада является относительно слабый профиль плотности мощности в силу продолжительности периода полураспада. С другой стороны, они, подобно ионисторам, способны, при необходимости, обеспечить мгновенную мощность. Последней проблемой радиоактивных источников является их значительный вес, в большей степени обусловленный массой свинца, необходимого для защиты от радиации. Например, для цезия-137 требуется свинцовый экран толщиной 80 мм/Вт, это не может не отражаться на их стоимости.

Резюме по хранилищам энергии и другим формам энергообеспечения

Как уже упоминалось ранее, выбор правильного источника питания имеет решающее значение. В табл. 3.3 ниже произведено краткое сравнение различных

системных факторов, которые должны учитываться при выборе источника питания.

Таблица 3.3. Сравнение различных системных факторов, которые должны учитываться при выборе источника питания

Категория	Литий-ионная батарея	Ионистор
Плотность энергии	200 кВт·ч/кг	8–10 кВт·ч/кг
Количество циклов перезарядки	Падение емкости после 100–1000 циклов	Не ограничено
Время разряда	1–10 часов	От миллисекунд до секунд
Температура эксплуатации	От –20 до +60 °C	От –40 до +85 °C
Рабочее напряжение	От 1,2 до 4,2 В	От 1 до 3 В
Изменение напряжения	Постоянное в течение срока службы	Линейное или экспоненциальное снижение
Скорость заряда	(очень медленно) 40 C/x	(Очень быстро) 5 C/x
Срок службы	От 0,5 до 5 лет	От 5 до 20 лет
Размеры	Очень малые	Большие
Стоимость	Невысокая ($250–1000)	Высокая ($10 000)

Заключение

В этой главе было рассмотрено несколько различных типов датчиков и оконечных точек, используемых при развертывании IoT. Интернет вещей не просто подключает устройство к интернету, хотя это ключевой момент, но суть IoT заключается в подключении аналогового мира к цифровому. Устройства и компоненты, ранее не подключенные к мировой сети, теперь имеют возможность делиться собранной информацией с внешним миром. Это грандиозно, поскольку теперь появилась возможность производить измерения, которые ранее никогда не производились. Возможность отслеживать и воспринимать внешний мир неизбежно ведет увеличению эффективности управления, а, следовательно, и к росту доходности и клиентской базы. Датчики позволяют умным городам производить взвешенное, интеллектуальное обслуживание, отслеживание активов и анализ огромного массива совокупных данных. Вопросы энергообеспечения проектируемых систем имеют ключевое значение, и это должно быть понятно проектировщикам. Плохо спроектированная система энергоснабжения приводит к слишком короткому времени автономной работы системы в целом, а это влечет за собой существенные затраты на исправление и доработку.

В следующей главе будут рассмотрены способы подключения оконечных точек к сети интернет без IP-протокола. Мы рассмотрим наиболее распространенные в пространстве IoT различные беспроводные персональные сети, их различия и функциональность.

Глава 4

Теория коммуникации и информации

Интернет вещей – это больше, чем просто данные от датчика. Мы должны сначала понять и сделать модель переноса данных движущихся датчиков из самых отдаленных мест на Земле в облако. Существует значительное количество технологий и путей передачи для перемещения данных, и основная часть материала в этой книге будет исследовать аспекты, ограничения и сравнения вариантов коммуникации для архитектора.

Мы начинаем обсуждение WAN с обзора беспроводных радиосигналов и факторов, влияющих на качество, ограничения, помехи, модели, пропускную способность и диапазон. В разных диапазонах есть много протоколов обмена WAN, и архитектор должен понимать плюсы и минусы выбора одного радиочастотного спектра по сравнению с другим.

Рисунок 4.1 помогает определить различные диапазоны и скорости передачи данных для беспроводных протоколов, которые мы рассмотрим в последующих главах. WPAN часто используется с другими акронимами для ближней связи, такими как **Field Area Network (FAN)**, **Wireless Local Area Network (WLAN)**, беспроводная **Home Area Network (HAN)**, **wireless Neighborhood Area Network (NAN)** и **Wireless Body Area Network (WBAN)**.

В этой главе будут представлены основополагающие модели и теория систем связи, частотных пространств и теории информации. Коммуникационные ограничения и модели сравнения будут предоставлены, чтобы позволить архитектору понять, как и почему работают определенные типы передачи данных и где они не будут работать.

Итак, мы начинаем с теории коммуникации, поскольку она играет фундаментальную роль в выборе правильного сочетания беспроводных технологий для развертывания решения IoT.

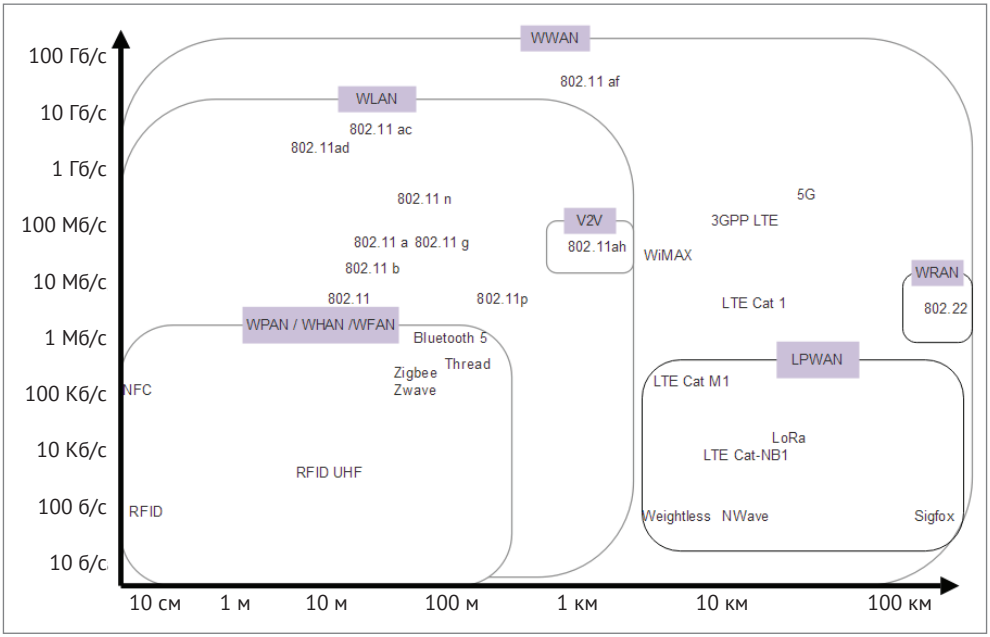


Рис. 4.1 ❖ Различные протоколы и категории беспроводной связи, предназначенные для разных диапазонов, скорости передачи данных и вариантов использования (мощность, транспорт и т. д.)

Теория коммуникации

Интернет вещей является конгломератом многих отдельных устройств, которые автономно производят и/или потребляют данные на очень далеком краю слоев сетей и протоколов. Важно понимать ограничения в построении систем связи для IoT или в любой форме сетей. Интернет вещей объединит персональные сети, локальные сети и дальние глобальные сети в сеть каналов связи. Большая часть того, что делает IoT возможным, строится вокруг коммуникационной основы; поэтому эта глава посвящена рассмотрению основ сетевых и коммуникационных систем. Теперь мы сосредоточимся на системах связи и сигнализации. Мы изучим диапазон, энергию и ограничения систем связи и то, как архитектор будет использовать эти инструменты для разработки успешного решения IoT.

Радиочастотная энергия и теоретический диапазон

Важно рассмотреть диапазон передачи, когда речь идет о беспроводных персональных сетях или любом радиочастотном протоколе беспроводной связи. Конкурирующие протоколы используют диапазон, скорость и мощность как разделители. В качестве архитекторов мы должны учитывать различные про-

токолы и варианты дизайна при реализации полного решения. Диапазон передачи основан на расстоянии между передатчиком и приемными антеннами, частотой передачи и мощностью передачи.

Наилучшая форма радиопередачи – это беспрепятственный прямой видимый участок без дополнительных радиосигналов. В большинстве ситуаций эта идеальная модель не будет существовать. В реальном мире есть препятствия, отражения сигналов, несколько беспроводных радиочастотных сигналов и шум.

При рассмотрении конкретной глобальной сети и сигнала меньшей скорости, например, 900 МГц по сравнению с несущим сигналом 2,4 ГГц, вы можете получить ослабление функции длины волны для каждой частоты. Это даст указания относительно мощности сигнала в любом диапазоне. Общая форма уравнения передачи Фрииса (рис. 4.2) помогает нам здесь:

$$P_r = P_t G_{Tx} G_{Rx} \frac{\lambda^2}{(4\pi R)^2}.$$

Децибельная (дБ) форма уравнения Фрииса определяется как:

$$P_r = P_t + G_{Tx} + G_{Rx} + 20\log_{10}\frac{\lambda}{(4\pi R)^2},$$

где G_{Tx} и G_{Rx} – коэффициент усиления передатчика и приемника, R – расстояние между передатчиком и приемником, а P_r и P_t – мощность приемника и передатчика соответственно.

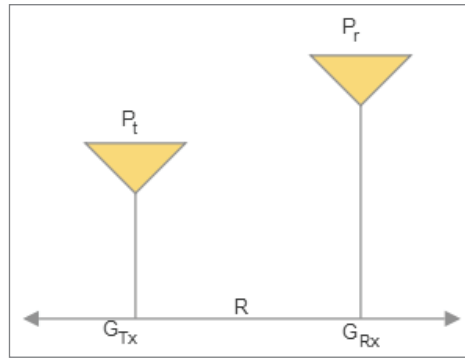


Рис. 4.2 ❖ Графическое представление уравнения Фрииса

Сигнал в 900 МГц на 10 м будет иметь потери 51,5 дБ, а сигнал 2,4 ГГц на 10 м будет иметь потерю 60,0 дБ.

Мы можем доказать, как мощность и диапазон влияют на качество сигнала, используя отношение, называемое *бюджетом ссылки*. Это сравнение мощности передачи с уровнем чувствительности, и оно измеряется по логарифмической шкале (дБ). Можно просто хотеть повысить уровень мощности для удовлетворения требований к диапазону, но во многих случаях это нарушает

нормативные требования или влияет на срок службы батареи. Другой вариант заключается в улучшении уровня чувствительности приемника, который точно соответствует Bluetooth 5 последней спецификации. Бюджет ссылки определяется соотношением мощности передатчика и чувствительности приемника, как показано ниже.

$$\text{БюджетСсылки} = \frac{\text{Мощность передачи } Tx}{\text{Уровень чувствительности } Sx}.$$

Бюджет связей измеряется по логарифмической шкале в дБ; поэтому добавление децибел эквивалентно умножению числовых коэффициентов, которые дают нам уравнение:

Мощность приемника (дБ) = Сила передачи (дБ) + Усиление (дБ) – Потери (дБ).

Предполагая, что нет никакого фактора, способствующего усилению сигнала (например, усиление антенны), есть только два способа улучшения приема: увеличение мощности передачи или уменьшение потерь.

Когда архитектор должен моделировать максимальный диапазон конкретного протокола, он будет использовать **Free-Space Path Loss (FSPL)**. Это величина потери сигнала электромагнитной волны по прямой видимости в свободном пространстве (без препятствий). Вторым фактором FSPL являются частота (f) сигнала, расстояние (R) между передатчиком и приемником и скорость света (c). В терминах вычисления FSPLF в децибелах уравнение будет:

$$\begin{aligned} FSPL\ (\text{дБ}) &= 10\log_{10}\left(\left(\frac{4\pi Rf}{c}\right)^2\right) \\ &= 20\log_{10}\left(\frac{4\pi Rf}{c}\right) \\ &= 20\log_{10}(R) + 20\log_{10}(f) + 20\log_{10}\left(\frac{4\pi}{c}\right) \\ &= 20\log_{10}(R) + 20\log_{10}(f) - 147{,}55. \end{aligned}$$

Формула FSPL представляет собой простой расчет первого порядка. Лучшее приближение учитывает отражения и волновые помехи от земной поверхности, такие как формула *потерь на плоской земле*. Здесь h_t – высота передающей антенны, h_r – высота приемной антенны. k представляет собой число волн в свободном пространстве и упрощается, как показано. Преобразуем уравнение для использования дБ-обозначения:

$$\frac{P_r}{P_t} = L_{\text{потерь на плоской земле}} \approx \left(\frac{\lambda}{4\pi R} k \frac{2h_t h_r}{R}\right) \approx \frac{h_t^2 h_r^2}{R^4},$$

где $k = \dfrac{2\pi}{\lambda}$.

То, что известно как потери на плоской земле, заключается в том, что расстояние влияет на потери на 40 дБ за декаду. Увеличение высоты антенны помогает. Типы помех, которые могут произойти естественным образом, включают:

- **отражения**: когда распространяющаяся электромагнитная волна натыкается на объект и приводит к множественным волнам;
- **дифракция**: когда радиоволновый путь между передатчиком и приемником затруднен объектами с острыми краями;
- **рассеяние**: когда среда, через которую проходит волна, состоит из объектов, чей размер меньше длины волны и количество таких препятствий велико.

Это важная концепция, поскольку архитектор должен выбрать решение WAN, частота которого уравновешивает пропускную способность данных, максимальный диапазон сигнала и способность сигнала проникать в объекты. Увеличение частоты, естественно, увеличивает потери в свободном пространстве (например, сигнал 2,4 ГГц имеет покрытие на 8,5 дБ меньше, чем сигнал 900 МГц). Вообще говоря, сигналы 900 МГц будут надежными на удвоенной дистанции сигналов 2,4 ГГц. Сигналы 900 МГц имеют длину волны 333 мм против 125 мм для сигнала 2,4 ГГц. Это позволяет сигналу с частотой 900 МГц обладать лучшей проникающей способность и не так сильно зависеть от рассеяния.

Рассеяние является важной проблемой для систем глобальной сети, так как многие развертывания не имеют прямой видимости между антеннами – вместо этого сигнал должен проникать сквозь стены и полы. Некоторые материалы вносят свой вклад в ослабление сигнала.

Потери на 6 дБ равны 50%-ному снижению мощности сигнала, а потеря на 12 дБ равна 75%-ному снижению.

Мы видим, что 900 МГц имеет преимущество перед 2,4 ГГц при проникновении через материал (см. табл. 4.1 и рис. 4.3).

Таблица 4.1. Материалы и потери радиосигнала

Материал	Потери в дБ на 900 МГц	Потери в дБ на 2,4 ГГц
Стекло 6 мм	–0,8 дБ	–3 дБ
Кладка из кирпичей или кирпичных блоков (20 см)	–13 дБ	–15 дБ
Гипсокартон	–2 дБ	–3 дБ
Дверь из цельного дерева	–2 дБ	–3 дБ

Как мы увидим, многие протоколы коммерчески доступны и используются во всем мире в спектре 2,4 ГГц. 2,4 ГГц обеспечивает пропускную способность в пять раз больше по сравнению с сигналом 900 МГц и может иметь гораздо меньшую антенну. Кроме того, спектр 2,4 ГГц нелицензирован и доступен для использования во многих странах. Сравнение частот показано в табл. 4.2.

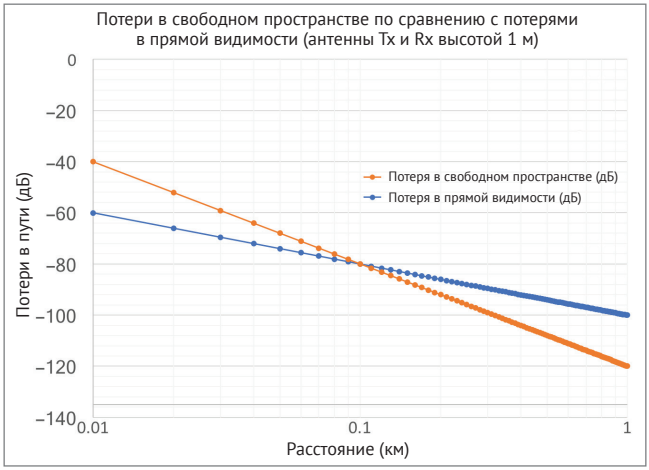


Рис. 4.3 ❖ Потеря в свободном пространстве по сравнению с плоской землей (в дБ) с использованием сигнала 2,4 ГГц с антеннами высотой 1 метр

Таблица 4.2. Сравнение частот 900 МГц и 2,4 ГГц

	900 МГц	**2,4 ГГц**
Сила сигнала	В основном, надежен	Переполненный диапазон, подверженный интерференции
Расстояние	В 2,67x дальше, чем 2,4 ГГц	Короче, но может компенсировать это улучшенным кодированием (Bluetooth 5)
Проникновение	Большая длина волны позволяет проникать через большинство материалов и препятствий	Потенциально подвержено влиянию большинства строительных материалов
Скорость передачи данных	Ограничена	От 2-х до 3-х раз быстрее, чем 900 МГц
Влияние на сигнал	Сигнал может зависеть от высоких объектов и препятствий, лучше проходит через листву	Меньшая вероятность взаимодействия канала с определенными объектами
Влияние на канал	Интерференция с беспроводными телефонами 900 МГц, сканерами RFID, сигналами соты, мониторами для детей	Интерференция с 802.11 Wi-Fi
Стоимость	Средняя	Низкая

Эти уравнения дают теоретическую модель; никакие аналитические уравнения не дают точного предсказания для определенных реальных сценариев, таких как многопутевые потери.

Радиочастотная интерференция

В этой главе мы увидим несколько новых схем снижения интерференции сигналов. Это проблема для многих форм беспроводной технологии, поскольку спектр

нелицензированный и общий (об этом мы поговорим подробнее в следующем разделе). Из-за того, что может быть несколько устройств, излучающих радиочастотную энергию в общем пространстве, будет возникать интерференция.

Возьмите Bluetooth и 802.11 Wi-Fi; оба работают в общем спектре 2,4 ГГц, но остаются работоспособными даже в перегруженных средах. Протокол **Bluetooth Low Energy (BLE)**, как мы увидим, будет случайным образом выбирать один из каналов 40–2 МГц в качестве формы скачкообразного изменения частоты. Мы видим на рис. 4.4 одиннадцать бесплатных каналов (три из которых рекламируются) на BLE, которые имеют 15% вероятность коллизий (тем более, что 802.11 не перескакивает между каналами). Новая спецификация Bluetooth 5 предоставляет такие методы, как маски доступа к слотам для блокировки областей Wi-Fi из списка частотных переходов. Существуют и другие методы, которые мы рассмотрим позже. Здесь мы показываем диапазон ILM для Zigbee и Bluetooth Low Energy. Также показано возможное соперничество с тремя каналами Wi-Fi в спектре 2,4 ГГц.

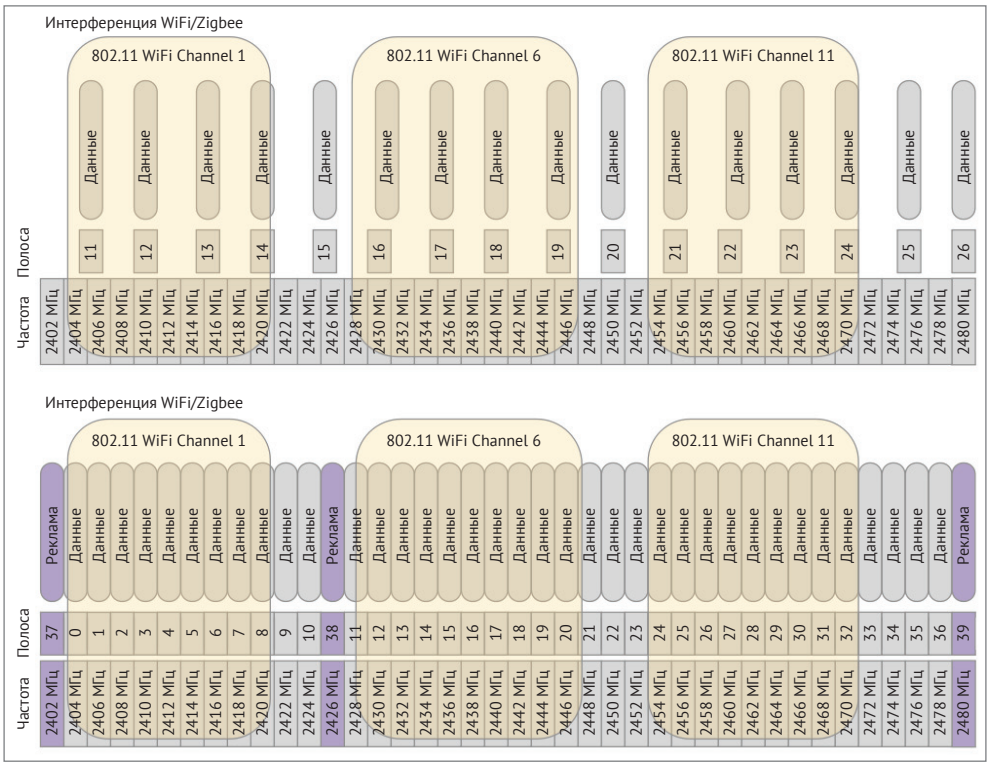


Рис. 4.4 ❖ Сравнение Bluetooth Low Energy (BLE) и Zigbee Interference с 802.11 Wi-Fi-сигналами в диапазоне 2,4 ГГц. BLE обеспечивает большее количество слотов и скачкообразную перестройку частоты для связи в случае коллизий Wi-Fi

Теория информации

Существуют предварительные теории, которые необходимо понимать, прежде чем детализировать спецификацию WAN. Две области, которые связаны с коммуникацией, – это то, как битрейт влияет на мощность передачи, что, в свою очередь, влияет на диапазон. Как мы узнаем, существуют ограничения целостности данных и битрейтов. Кроме того, нам необходимо классифицировать узкополосную и широкополосную связь.

Пределы битрейта и теорема Шеннона-Хартли

В коммуникациях на дальнем расстоянии и ближней связи цель заключается в максимизации битрейта и расстояния в пределах ограничений спектра и шума. Теорема Шеннона-Хартли состоит из работы Клода Шеннона из Массачусетского технологического института в 1940-х гг.[1] и Ральфа Хартли из Bell Labs в 1920-х гг.[2] Фундаментальная работа была сделана Гарри Найквистом, также Bell Labs, который определил максимальное количество импульсов (или бит), которые могли перемещаться по телеграфу за единицу времени[3].

По сути, Найквист разработал предел выборки, который определяет, сколько теоретической ширины полосы пропускания имеет заданная частота дискретизации. Это называется нормой Найквиста и показано в следующем уравнении:

$$f_p \leq 2B.$$

Здесь f_p – частота импульсов, а B – ширина полосы в герцах. Это означает, что максимальный битрейт ограничен вдвое большей частотой дискретизации. Рассматривая это по-другому, уравнение идентифицирует минимальный битрейт, при котором требуется выборка сигнала конечной полосы пропускания, чтобы сохранить всю информацию. Отмена дискретизации приводит к эффекту сглаживания и искажению.

Затем Хартли разработал способ количественной оценки информации в так называемой скорости линии. Скорость линии можно рассчитать как биты в секунду (например, Мбит/с). Это известно как закон Хартли, и он является предшественником теоремы Шеннона. Закон Хартли просто утверждает, что максимальное количество различимых амплитуд импульсов, которые могут передаваться надежно, ограничено динамическим диапазоном сигнала и точностью, с которой приемник может точно интерпретировать каждый отдельный сигнал. Показан закон Хартли в терминах M (количество уникальных

---

[1] *Shannon C. E.* (1949/1998). Математическая теория коммуникации. Urbana, IL: University of Illinois Press.

[2] *Hartley R. V. L.* (июль, 1928 г.). Передача информации // Технический журнал Bell System.

[3] *Nyquist H.* Некоторые темы теории телеграфной передачи // Операции Американского института инженеров-электриков. 1928. Апр. Т. 47. № 2. С. 617–644.

форм амплитуды импульса), что эквивалентно соотношению количества напряжения:

$$M = 1 + \frac{A}{\Delta V}.$$

Преобразование уравнения в логарифм по основанию 2 дает нам скорость линии R:

$R = f_p \log_2(M)$.

Если мы объединим это с предыдущей нормой Найквиста, мы получим максимальное количество импульсов, которые могут быть переданы по одному каналу пропускной способности B. Однако Хартли не занимался точностью; на значение M (количество отдельных импульсов) может влиять шум.

$R \leq 2B \log_2(M)$.

Шеннон усилил уравнение Хартли, рассматривая эффекты гауссовского шума, и завершил уравнение Хартли отношением сигнал/шум. Шеннон также представил концепцию кодирования с исправлением ошибок вместо использования индивидуально различимых амплитуд импульсов. Это уравнение теперь известно как теорема Шеннона-Хартли:

$$C = B \log_2 \left(1 + \frac{S}{N}\right).$$

Здесь C – емкость канала в битах в секунду, B – полоса пропускания канала в герцах, S – средний принятый сигнал, измеренный в ваттах, а N – средний шум на канале, измеренный в ваттах. Эффект этого уравнения является небольшим, но важным. Для каждого уровня увеличения шума к сигналу в децибелах мощность резко падает. Аналогичным образом, улучшение отношения сигнал-шум увеличит пропускную способность. Без шума мощность была бы бесконечной.

Также возможно улучшить теорему Шеннона-Хартли, добавив в уравнение множитель n. Здесь n представляет собой дополнительные антенны или трубы. Мы рассмотрели это ранее как технологию **множественного ввода, множественного вывода (MIMO)**.

$$C = B \times n \times \log_2 \left(1 + \frac{S}{N}\right).$$

Чтобы понять, как правило Шеннона относится к ограничениям беспроводных систем, упомянутых в этой книге, нам нужно выразить уравнение с точки зрения энергии на бит, а не **соотношение сигнал/шум (SNR)**. Полезным примером на практике является определение минимального SNR, необходимого для достижения определенного битрейта. Например, если мы хотим передать $C = 200$ кбит/с по каналу с пропускной способностью B = 5000 кбит/с, тогда минимальное SNR дается как:

$$C = B\log_2\left(1+\frac{S}{N}\right).$$

$$200 = 500\times\log_2\left(1+\frac{S}{N}\right).$$

$$\frac{S}{N} = 0,028.$$

$$\frac{S}{N} = -15,528 \text{ дБ}.$$

Это показывает, что можно передавать данные с использованием сигнала, который слабее фонового шума.

Тем не менее, существует предел скорости передачи данных. Чтобы показать эффект, пусть E_b представляет энергию одного бита данных в джоулях. Пусть N_0 представляет спектральную плотность шума в ваттах/герцах. E_b/N_o представляет собой безразмерную единицу (как правило, выраженную в дБ), которая представляет SNR на бит, или широко известную как **энергоэффективность**. Выражения энергоэффективности устраняют смещения методов модуляции, кодирования ошибок и ширины полосы сигнала из уравнения. Предположим, что система совершенна и идеальна, где $R_B = C$, где R – пропускная способность. Теорема Шеннона-Хартли может быть переписана как:

$$\frac{C}{B} = \log_2\left(1+\frac{E_bC}{N_0B}\right).$$

$$\frac{E_b}{N_0} = \frac{2^{\frac{C}{B}}+1}{\frac{C}{B}}.$$

$$\frac{E_b}{N_0} \geq \lim_{\frac{C}{B}\to 0}\frac{2^{\frac{C}{B}}-1}{\frac{C}{B}} = \ln(2) = 1,59 \text{ дБ}.$$

Это известно как **предел Шеннона** для **аддитивного белого гауссовского шума (AWGN)**. AWGN является каналом и просто базовой формой шума, обычно используемой в теории информации для выражения эффектов случайных процессов в природе. Эти источники шума всегда присутствуют в природе и включают такие вещи, как тепловые колебания, излучение черного тела и остаточные эффекты Большого Взрыва. «Белый» аспект шума подразумевает равное количество шума, добавляемого к каждой частоте. Предел можно нарисовать на графике, показывающем спектральную эффективность по сравнению с SNR на бит и показанном на рис. 4.5.

Области, представляющие интерес на рис. 4.5, включают $R > B$ «Невозможная область». Эта область выше предела Шеннона для кривой. Он говорит, что

надежная форма обмена информацией не может быть выше предельной. Область ниже предела Шеннона называется «реализуемым регионом», где $R < B$. Каждый протокол и технология модуляции в любой форме коммуникации пытаются приблизиться как можно ближе к пределу Шеннона. Мы можем видеть, где располагается типичный 4G LTE с использованием различных форм модуляции.

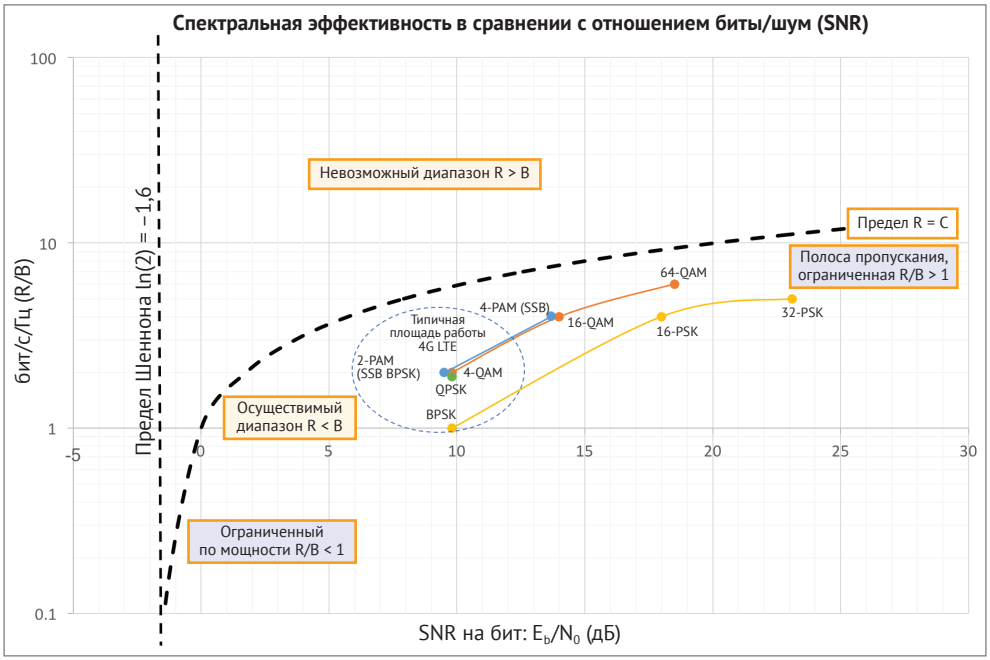


Рис. 4.5 ❖ Кривая спектральной эффективности от SNR (энергетическая эффективность). Пунктирная линия представляет собой предел Шеннона, который сходится при ln (2) = –1,6. Различные схемы модуляции показаны в пределе Шеннона с типичным диапазоном сигналов 4G LTE

Есть еще две области, представляющие интерес. Область «Полоса пропускания ограничена» в направлении направо наверх допускает высокую спектральную эффективность и хорошие значения SNR E_B/N_O. Единственным ограничением в этом пространстве является компромисс с фиксированной или санкционированной спектральной эффективностью против неограниченной мощности передачи P, что означает, что емкость значительно выросла по доступной полосе пропускания. Противоположный эффект называется областью ограниченной мощности в направлении нижнего левого угла диаграммы. Область «Мощность ограничена» – это та, где SNR E_B/N_O очень низкое, поэтому предел Шеннона приводит нас к низким значениям спектральной эффектив-

ности. Здесь мы жертвуем спектральной эффективностью для получения заданного качества передачи *P*.

Примером ограниченного использования энергии является космический аппарат, такой как зонд Кассини, отправленный к Сатурну. В этом случае потеря сигнала в пространстве чрезвычайно велика, и единственным способом получить надежные данные является снижение скорости передачи данных до очень низких значений. Мы также видим это в Bluetooth 5, используя новый BLE Coded PHY. В этом случае скорость Bluetooth снижается с 1 или 2 Мбит/с до 125 Кбит/с для улучшения диапазона и целостности данных.

На рис. 4.5 также показаны некоторые типичные схемы модуляции, используемые сегодня, такие как фазовый сдвиг, QAM и другие. Предел Шеннона также показывает, что произвольное улучшение метода модуляции, такого как квадратурная амплитудная модуляция для 4-QAM до 64-QAM, не масштабируется линейно. Преимущество более высоких порядков модуляции (например, 4-QAM против 64-QAM) заключается в том, что вы можете передавать больше бит на символ (два против шести). Основным недостатком более высоких порядков модуляции является:

- использование модуляции более высокого порядка требует большего SNR для работы;
- более высокие порядки модуляции требуют гораздо более сложных схем и алгоритмов DSP, увеличивающих сложность;
- увеличение передачи бит на символ увеличивает коэффициент ошибок.

Теорема Шеннона утверждает, что существует максимальная скорость информации, которая может передаваться по каналу связи при наличии аддитивного гауссовского белого шума. По мере уменьшения шума скорость информации будет увеличиваться, но имеет конечную границу, которая не может быть нарушена. В любой ситуации, если скорость передачи R меньше пропускной способности канала C, тогда должен быть метод или технология для передачи данных без ошибок.

Частота битовых ошибок

Другой важной характеристикой передачи данных является **частота битовых ошибок (BER)**. BER показывает число битовых ошибок, полученных по каналу связи. BER – это безразмерное измерение, выраженное как отношение или процент. Например:

Если начальная последовательность такова: 1 0 1 0 1 1 0 1 0 0,
а полученная последовательность такова: **0** 0 1 0 1 **0 1 0 1** 0 (различия выделены жирным),
тогда BER будет 5 ошибок/10 переданных бит = 50%.

На BER влияет шум канала, интерференция, замирание вследствие многолучевости и затухание. Методы улучшения BER включают в себя увеличение мощности передачи, улучшение чувствительности приемника, использование методов модуляции менее плотного / более низкого порядка или добавление

избыточных данных. Последний метод обычно называют **прямой коррекцией ошибок (FEC)**. FEC просто добавляет дополнительную информацию к передаче. В самом основном смысле, можно было бы добавить тройную избыточность и алгоритм выбора большинства; однако это уменьшит пропускную способность в 3 раза. Современные методы FEC включают коды помех и коды ошибок с ошибкой Рида-Соломона. BER может быть выражен как функция SNR E_b/N_o.

На рис. 4.6 показаны различные методы модуляции и их соответствующие BER для различных SNR.

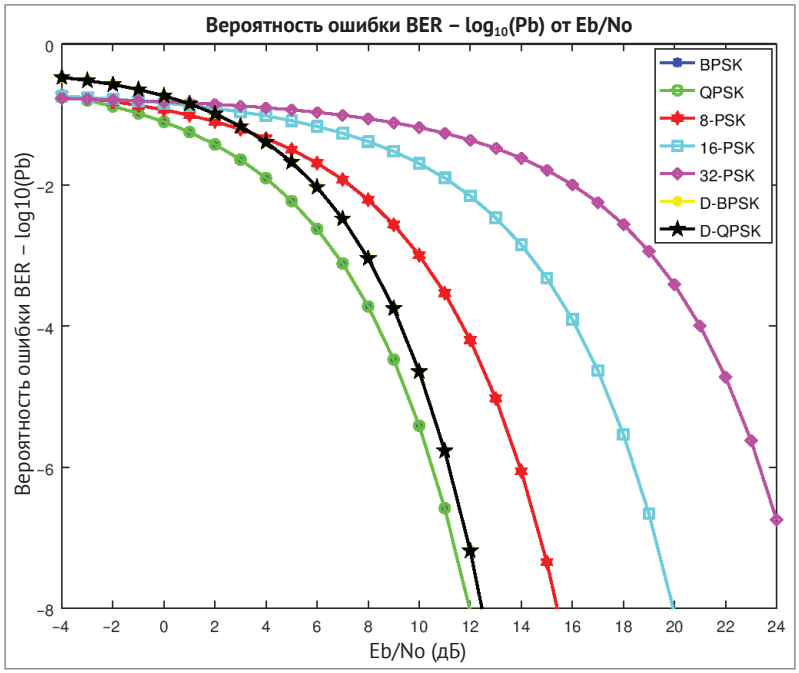


Рис. 4.6 ❖ Частота битовых ошибок (Pb) по сравнению с энергоэффективностью (E_b/N_o) SNR для различных схем модуляции. По мере того как SNR увеличивается вправо, BER естественно уменьшается

На этом этапе нужно понимать следующее:

- теперь мы можем рассчитать минимальное SNR, необходимое для достижения определенной скорости передачи данных для системы;
- единственный способ увеличить емкость или пропускную способность для беспроводной службы – это:
  - добавить больше спектра и емкости канала, что увеличивает полосу пропускания линейно;
  - добавить больше антенн (MIMO), что линейно улучшает полосу пропускания;

  - улучшить SNR с помощью усовершенствованных антенн и приемников, что только улучшает уравнение логарифмически;
- предел Шеннона является конечной границей цифровой передачи; превышение предела возможно, но целостность данных будет потеряна;
- факторы, которые вносят вклад в шум;
- нельзя просто увеличить уровни модуляции, не увеличивая затраты на ошибки и сложность.

Для сотовых сигналов 4G LTE, которые рассматриваются ниже в этой книге, он работает в диапазоне от 700 МГц до 5 ГГц с десятками сегрегированных полос в этом диапазоне. Сотовый телефон (или устройство IoT на основе батареи) имеет значительно меньшую мощность, чем башня сотовой связи, но часто бывает, что устройство IoT будет передавать данные датчика в облако. Восходящая линия от устройства IoT – это то, что мы рассматриваем здесь. Предел мощности восходящей линии связи составляет 200 мВт, что составляет 23 дБм. Это ограничивает общий диапазон передачи. Этот предел, однако, является динамическим и будет зависеть от ширины полосы канала и скорости передачи данных. 4G, как и несколько устройств WPAN и WLAN, используют мультиплексирование с ортогональным частотным разделением. Каждый канал имеет множество поднесущих для решения проблем многолучевого замирания. Если вы суммируете все данные, передаваемые по поднесущим, вы достигаете высоких скоростей передачи данных.

4G LTE обычно использует каналы 20 МГц, а LTE-A может использовать каналы 100 МГц. Эти широкие каналы ограничены общим спектром доступности и конкурируют с несколькими операторами (ATT, Verizon и т. д.) и другими технологиями, использующими спектр. Дополнительной сложностью сотовой связи является то, что носитель может иметь части спектра, разделенные и не пересекающиеся друг с другом.

Cat-3 LTE может использовать каналы 5, 10 или 20 МГц. Самая маленькая зернистость канала составляет 1,4 МГц. LTE-A позволяет объединять до пяти каналов с частотой 20 МГц для совокупной полосы пропускания 100 МГц.

Метод измерения расстояния, на котором работает беспроводное устройство, – это **максимальная потеря связи (MCL)**. MCL – это максимальное расстояние, на котором происходит полная потеря канала между передатчиком и приемной антенной, но услуга передачи данных все еще может быть обеспечена. MCL – очень распространенный способ измерения охвата системы. MCL будет включать в себя усиление антенны, потерю пути, затенение и другие радиоэффекты. Как правило, система 4G LTE будет иметь MCL около 142 дБ. Мы рассмотрим MCL при исследовании сотовых технологий IoT, таких как Cat-M1.

На этом этапе мы должны понять, что, если мы увеличим время прослушивания на каждый бит, уровень шума снизится. Если мы уменьшим битрейт в 2 раза, тогда будет верно следующее: (битрейт/2) = (протяженность бита × 2). Кроме того, энергия на бит увеличивается в 2 раза, а энергия шума увеличивается на sqrt (2). Например, если мы уменьшаем Bit_Rate с 1 Мбит/с до 100 кбит/с, тогда Bit_Duration = увеличивается в 10 раз. Диапазон улучшается на sqrt (10) = 3,162x.

Узкополосная и широкополосная связь

Многие беспроводные протоколы, которые мы рассмотрим, известны как широкополосные. Мы увидим, что альтернативная (узкополосная) связь также имеет место, особенно для LPWAN. Различия между узкополосной и широкополосной сетью заключаются в следующем:

- **узкополосная связь**: радиоканал, пропускная способность которого не превышает пропускную способность когерентности канала. Как правило, когда мы говорим об узкополосной связи, мы говорим о сигналах, которые имеют пропускную способность 100 кГц или меньше. В узкополосной многолучевой передаче амплитуда и фаза изменяются. Узкополосный сигнал будет исчезать равномерно, поэтому добавление большего количества частот не приносит пользы сигналу. Узкополосные каналы также называются плоскими замирающими каналами, потому что они обычно передают все спектральные компоненты с равным коэффициентом усиления и фазой друг к другу;
- **широкополосный канал**: радиоканал, пропускная способность которого может значительно превышать когерентную полосу пропускания. Они, как правило, превышают 1 МГц в полосе пропускания. Здесь многолучевое распространение вызывает проблему с самоинтерференцией. Широкополосные каналы также называются частотно-избирательными, поскольку разные части общего сигнала будут зависеть от разных частот в широкополосной связи. Вот почему широкополосные сигналы используют широкий диапазон частот для распределения мощности во многих диапазонах когерентности для уменьшения эффектов замирания.

Время когерентности является мерой минимального времени, требуемого для изменения амплитуды или фазы, чтобы стать некоррелированными с предыдущим значением.

Мы рассмотрели некоторые формы эффектов затухания; однако их гораздо больше. Потеря по пути – типичный случай, когда потеря пропорциональна расстоянию. Тень – это то место, где ландшафт, здания и холмы создают препятствия сигналу по сравнению с свободным пространством, а затухание многолучевости происходит при рекомбинированном рассеянии и волновой интерференции радиосигнала на объектах (из-за дифракции и отражений). Другие потери включают допплеровские сдвиги, если радиочастотный сигнал находится в движущемся транспортном средстве. Существуют две категории явления затухания:

- **быстрое замирание**: это характерно для многолучевого замирания, когда время когерентности невелико. Канал будет меняться каждые несколько символов; поэтому время когерентности будет низким. Этот тип замирания также известен как затухание Рэлея, что является вероятностью случайных дисперсий, которые вызывают радиочастотный сигнал из-за атмосферных частиц или сильно застроенных мегаполисов;

- **медленное замирание**: это происходит, как правило, из-за доплеровского распространения или затенения, когда время когерентности большое и происходит движение на большие расстояния. Здесь время когерентности достаточно длинное, чтобы успешно передавать значительно больше символов, чем быстрый путь замирания.

На рис. 4.7 показана разница между быстрыми и медленными путями замирания.

Рис. 4.7 ❖ Различные эффекты затухания радиочастотного сигнала. Слева направо: общие потери по пути на прямой видимости. Средний: эффекты медленного замирания из-за больших структур или ландшафтов. Справа: комбинированные эффекты расстояния, медленного замирания и быстрого замирания

Мы увидим, что технологии, использующие узкополосные сигналы, используют так называемое **временное разнесение** для преодоления проблем с быстрым угасанием. Разнесение по времени просто означает, что сигнал и полезная нагрузка передаются несколько раз в надежде, что одно из сообщений пройдет.

В многопутевом сценарии разброс задержек – это время между импульсами из разных многолучевых сигналов. В частности, это задержка между первым поступлением сигнала и самым ранним приходом многолучевой составляющей сигнала.

Когерентная полоса пропускания определяется как статистический диапазон частот, в которых канал считается плоским. Это период времени, когда две частоты, вероятно, будут иметь сопоставимое затухание. Ширина когерентной полосы B_c обратно пропорциональна разбросу задержки D:

$$B_C \approx \frac{1}{D}.$$

Время, в течение которого символ может передаваться без межсимвольной интерференции, равно $1/D$. На рис. 4.8 показана когерентная полоса пропускания для узкополосной и широкополосной связи. Поскольку широкополосная связь больше, чем ширина полосы когерентности B_c, она, скорее всего, имеет независимые атрибуты затухания. Это означает, что разные частотные компоненты будут иметь некоррелированное замирание. В то время, как узкополос-

ные частотные компоненты все вписываются в B_c и будут испытывать равномерное замирание.

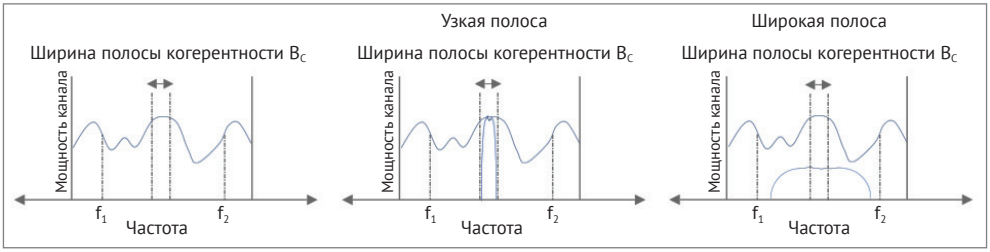


Рис. 4.8 ❖ Полоса пропускания когерентности и влияние на узкополосную и широкополосную связь. Частоты f_1 и f_2 будут замирать независимо, если $|f_1 - f_2| B_c$. Очевидно, что узкополосная линия находится внутри B_c, и широкополосная видимость явно превосходит диапазон B_c с некоторым запасом

Необходимо обеспечить, чтобы время между отправкой нескольких сигналов из многолучевого сценария было разделено достаточно, чтобы не мешать символам. Это называется **Inter-Symbol Interference (ISI)**. На рис. 4.9 показано, что задержки слишком коротки, чтобы вызвать ISI. Учитывая, что общая ширина полосы $B >> 1/T$ (где T – время ширины импульса) и $B1/D$ испытывает влияние, мы можем тогда вообще заявить, что ширина полосы пропускания должна быть намного больше ширины когерентности: $B >> B_c$.

Как правило, более низкие частоты обладают большей проникающей способностью и меньшими помехами, но требуют больших антенн и имеют меньшую доступную полосу пропускания для передачи. Более высокие частоты имеют большую потерю по пути, но меньшие антенны и большую пропускную способность.

На общий битрейт будет влиять разброс задержки. Например, предположим, что мы используем модуляцию QPSK, а BER = 10^{-4}, тогда для различных разбросов задержки (D) имеем:

- D = 256мкс: 8 КБ/с;
- D = 2.5 мкс: 80 КБ/с;
- D = 100 нс: 2 Мб/с.

Радиоспектр

Беспроводная связь основана на радиоволнах и диапазонах частот в общем радиочастотном спектре. В следующей главе мы рассмотрим связь на дальнем расстоянии для сотовых и других сред дальней связи, здесь мы сосредоточимся на диапазоне 1000 метров или меньше. Здесь мы рассмотрим процесс распределения спектра, а также типичные частоты использования WAN-устройств.

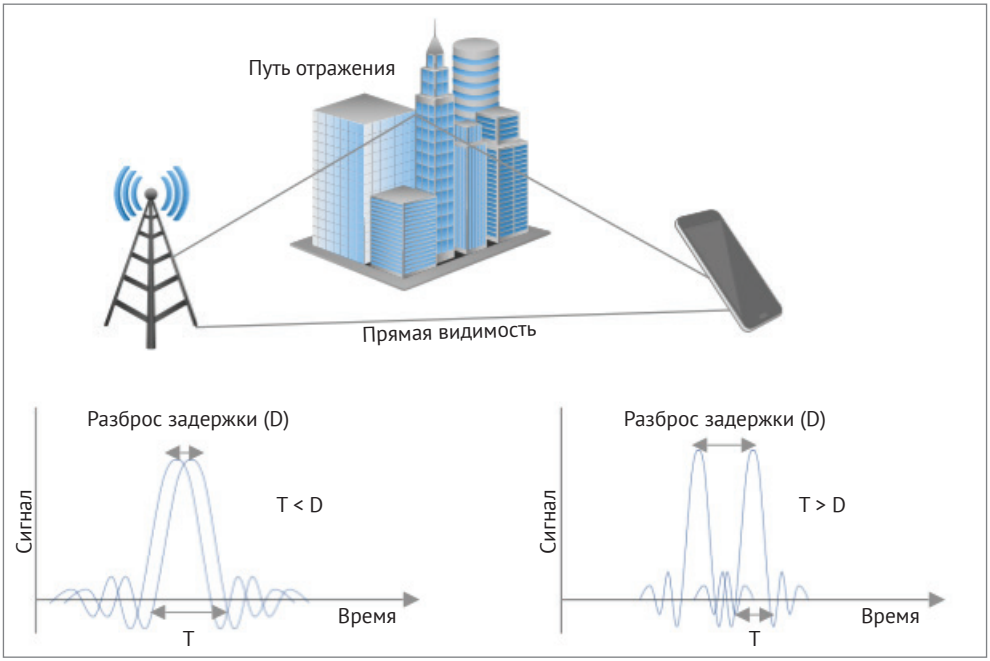


Рис. 4.9 ❖ Пример распространения задержки.
Здесь 2 сигналит из формы многолучевого события. Если разброс *D* задержки меньше, чем ширина импульса *T*, то сигналы могут не распространяться достаточно далеко, чтобы перебить другой компонент многолучевого распространения. Если задержка распространения достаточно велика, конфликт между многолучевым распространением невозможен

Управляющая структура

Спектр варьируется от 3 Гц до 3 ТГц, а распределение в пределах спектра регулируется **International Telecommunication Union (ITU)**. Полоса считается частью спектра, которая может быть выделена, лицензирована, продана или свободно использована в зависимости от частоты. С точки зрения ITU полосы классифицируются следующим образом (табл. 4.3).

В Соединенных Штатах **Федеральная комиссия по связи (FCC)** и **Национальное управление связи и информации (NTIA)** контролируют права на использование спектра частот. FCC управляет использованием не федерального спектра, в то время как NTIA управляет федеральным использованием (армия, FAA, ФБР и т. д.).

Полный спектр, управляемый FCC, варьируется от КГц и до частот ГГц включительно. Общее распределение и принадлежность частот показаны на рис. 4.10. Выделены интересные частоты, которые будут обсуждаться в этой книге.

Таблица 4.3. Классификация частот согласно ITU

Частота	Полоса IEEE	ЕС, НАТО, ECM США	ITU	
			полоса	аббревиатура
03 Гц				
3 Гц		A	1	ELF
30 Гц			2	SLF
300 Гц			3	ULF
3 кГц			4	VLF
30 кГц			5	LF
300 кГц			6	MF
3 МГц	HF		7	HF
30 МГц	VHF		8	VHF
250 МГц		B		
300 МГц	UHF		9	UHF
500 МГц		C		
1 ГГц	L	D		
2 ГГЦ	S	E		
3 ГГц		F	10	SHF
4 ГГц	C	G		
6 ГГц		H		
8 ГГц	X	I		
10 ГГц		J		
12 ГГц	Ku			
18 ГГц	K			
20 ГГц		K		
27 ГГц	Ka			
30 ГГц			11	EHF
40 ГГц	V	L		
60 ГГц		M		
75 ГГц	W			
100 ГГц				
110 ГГц	mm			
300 ГГц			12	THF
3 ТГц				

На рис. 4.11 показана небольшая часть распределения частот в диапазоне 900 МГц – 2,7 ГГц (общая для сигналов WPAN) и как распределяются и распространяются частоты в настоящее время. Во многих областях использование является многоцелевым и общим.

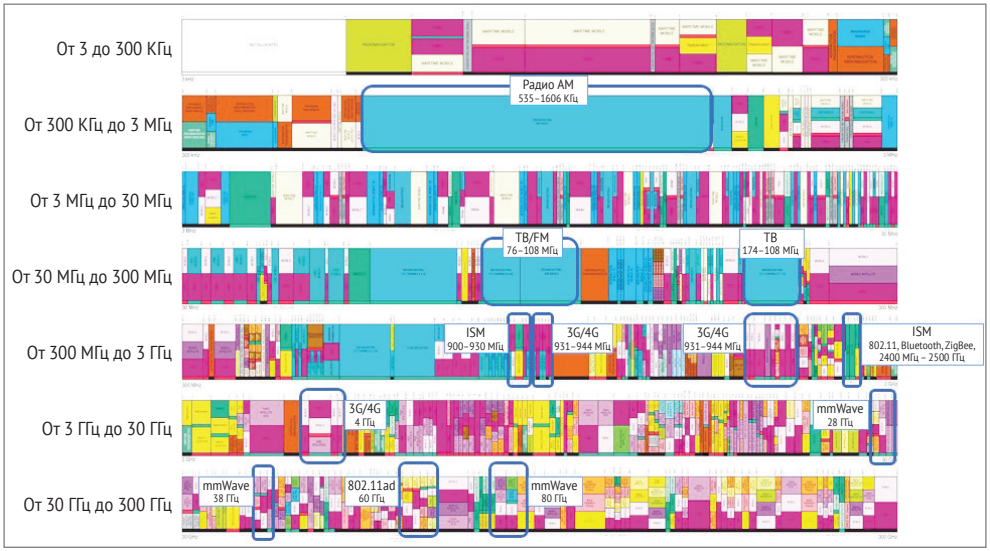


Рис. 4.10 ❖ Полный спектр распределения частот FCC с выделенными диапазонами, охваченными в этой книге

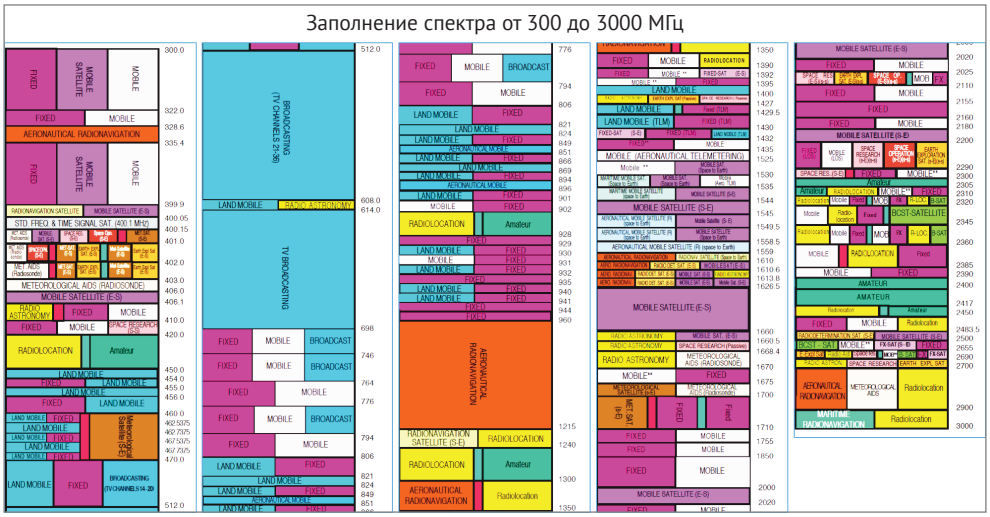


Рис. 4.11 ❖ Распределение частот FCC и NTIA между 300 МГц и 3 ГГц. Диаграмма представляет собой небольшую часть общего распределения частот. *Источник*: FCC, «Распределение радиочастот в США», октябрь, 2003 г.

FCC также назначает частоты в лицензированных и нелицензированных спектрах. В зонах «нелицензированный» или «лицензионно освобожденный» пользователи могут работать без лицензии FCC, но должны использовать сертифицированное радиооборудование и соответствовать техническим требованиям, таким как ограничения мощности и рабочие циклы. Они подробно описаны в документе «Правила FCC. Часть 15». Пользователи могут работать в этих спектрах, но будут подвержены интерференции.

Лицензионные зоны спектра позволяют использовать исключительный диапазон для определенных областей/местоположений. Выделение может предоставляться на национальной основе или в отдельных сегментах по месту. С 1994 г. эксклюзивность и права на эти области спектра были выставлены на аукционах для определенных областей/сегментов/рынков (например, рынки сотовой связи, экономические районы и т. д.). Некоторые группы могут быть гибридом двух моделей, где полосы, возможно, были лицензированы по принципу «одно место за другим», а более поздние группы, окружающие эти лицензии, были проданы на аукционе для более крупных географических или национальных районов. FCC также допускает вторичный рынок и устанавливает политику и процедуры посредством лизинга спектра и передачи контроля.

Развертывания IoT обычно будут использовать лицензированные регионы для связи на большие расстояния, что описано в следующей главе. Нелицензионный спектр обычно используется для **промышленных, научных и медицинских (ISM)** устройств. Для протоколов IoT, IEEE 802.11 Wi-Fi, Bluetooth и IEEE 802.15.4 все находится в нелицензированном спектре 2,4 ГГц.

Заключение

В этой главе предоставлен базовый материал для понимания теории и ограничений беспроводной связи. Более подробную информацию и более глубокие исследования рекомендуется изучить для понимания ограничений передачи данных второго и третьего порядка. Архитектор должен понимать разные модели и ограничения беспроводных сигналов, дисперсию радиочастотной энергии, диапазон и фундаментальные пределы теории информации, предоставляемые законом Шеннона. Архитектор должен также понимать стратегию управления и распределения частот. В этой главе также содержится терминология, которая будет повторно использоваться в следующих главах WPAN, WLAN и WAN.

В следующей главе начнется путешествие данных IoT от датчика к облаку через первый прыжок и через личную сеть близкого диапазона. Мы построим оттуда доступ к WLAN- и WAN-системам.

Глава 5

Беспроводная персональная сеть (WPAN) не на основе IP

Датчики и другие устройства, подключенные к интернету, нуждаются в способе передачи и получения информации. Это тема **персональной сети региона (PAN)** и ближней связи. В экосистеме интернета вещей связь с датчиком или исполнительным механизмом может осуществляться по медным проводам или **беспроводной персональной сетью региона (WPAN)**. В этой главе мы концентрируемся на WPAN, поскольку это распространенный метод промышленного, коммерческого и потребительского подключения к вещам в интернете. Связь на основе проводов по-прежнему используется, но, прежде всего, в унаследованных отраслях и областях, которые не являются благоприятными для радиосигналов. Существует множество различных каналов связи между конечной точкой и интернетом; некоторые могут быть построены на традиционном стеке IP (6LoWPAN), а другие используют не-IP (интернет-протокол) для максимизации экономии энергии (BLE).

Мы разделяем IP и не-IP, поскольку IP-системы связи нуждаются в дополнительной детализации, что связи не-IP не обязательно нужно. Системы связи, отличные от IP, оптимизированы для экономии затрат и энергии, тогда как решения на базе IP обычно имеют меньше ограничений (например, 802.111 Wi-Fi). В следующей главе будет описано перекрытие IP в WPAN и WLAN.

В этой главе будут рассмотрены некоммуникационные стандарты связи, различные топологии сетей WPAN (смешанная, звезда), а также ограничения и цели систем связи WPAN. Эти типы систем связи работают на расстоянии примерно от метра до 200 метров (хотя некоторые из них могут достичь гораздо большего). Мы углубимся в беспроводной протокол Bluetooth и новую спецификацию Bluetooth 5.0, поскольку он закладывает основу для понимания других протоколов и является распространенной и мощной частью решений IoT.

В эту главу войдут технические сведения о фирменных и открытых стандартах. Каждый протокол связи принят по определенным причинам и прецедентам; они также будут освещены в этой главе. Основные темы этой главы:

- качество и область распространения радиочастотного сигнала;
- распределение беспроводного спектра;
- беспроводной протокол Bluetooth с особым акцентом на новую спецификацию Bluetooth 5;
- 802.15.4;
- Zigbee®;
- Z-Wave®.

Стандарты беспроводной персональной локальной сети

В этом разделе будут рассмотрены три соответствующие беспроводные персональные сети в пространстве IoT. Большая часть этого раздела будет посвящена Bluetooth, поскольку он обеспечивает значительное количество функций и имеет очень глубокое влияние в экосистеме IoT. Кроме того, в Bluetooth 5.0 добавлено множество функций и возможностей, которые ранее не наблюдались в спецификации Bluetooth, что обеспечивает диапазон, мощность, скорость и возможности подключения, которые делают его самым мощным решением WPAN во многих случаях использования. Также будут исследованы сети Zigbee, Z-Wave и IEEE 802.15.4.

Также полезно знать, что термин WPAN перегружен. Первоначально он должен был быть буквальной средой и личной сетью для конкретного человека, связанного с переносимыми устройствами, но теперь его смысл расширился.

Стандарты 802.15

Многие протоколы и сетевые модели, описанные в этом разделе, базируются или основаны на рабочих группах IEEE 802.15. Первоначально группа 802.15 была создана таким образом, чтобы сосредоточиться на переносимых устройствах (эквивалент персональной сети). Их работа значительно расширилась и теперь фокусируется на более высоких протоколах скорости передачи данных, диапазонах измерений до километров и специальных коммуникациях. Более миллиона устройств отгружаются ежедневно с использованием протокола 802.15.x. Ниже приведен список различных протоколов, стандартов и спецификаций, которые IEEE поддерживает и регулирует:

- **802.15** – определения беспроводной локальной сети;
- **802.15.1** – оригинальная основа Bluetooth PAN;
- **802.15.2** – спецификации сосуществования для WPAN и WLAN для Bluetooth;
- **802.15.3** – высокая скорость передачи данных (55 Мбит/с и более) в WPAN для мультимедиа:
  - **802.15.3a** – высокоскоростные улучшения PHY;
  - **802.15.3b** – высокоскоростные усовершенствования MAC;

  - **802.15.3c** – высокоскоростная (> 1 ГБрs) передача с использованием технологии mm-wave (миллиметровая волна);
- **802.15.4** – низкая скорость передачи данных, простой, простой дизайн, многолетняя работа батарей (Zigbee):
  - **802.15.4-2011** – свод (спецификации a-c) включает в себя UWB, Китай и Японию PHY;
  - **802.15.4-2015** – свод (спецификация (d-p) включает поддержку RFID, медицински безопасную полосу PHY, низкую энергию, телевизионные белые пространства, железнодорожные связи;
  - **802.15.4r (в режиме ожидания)** – протокол распространения;
  - **802.15.4s** – использование ресурсов спектра (SRU);
  - **802.15.t** – высокая скорость PHY 2 Мбит/с;
- **802.15.5** – смешанная сеть;
- **802.15.6** – сетевая инфраструктура для медицины и развлечений;
- **802.15.7** – видимая световая связь с использованием структурированного освещения:
  - **802.15.7a** – расширяет диапазон до УФ и ближнего ИК-диапазона, с измененным именем «оптической беспроводной связи»;
- **802.15.8** – **Peer Aware Communications (PAC)** без доступа к сети со скоростью от 10 Кбит/с до 55 Мбит/с;
- **802.15.9** – протокол управления ключами (KMP), стандарт управления для обеспечения безопасности;
- **802.15.10** – маршрутизация сети уровня 2, рекомендует смешанную сеть для 802.15.4, multiPAN;
- **802.15.12** – интерфейс верхнего уровня, попытка упростить использование 802.15.4 для 802.11 или 802.3.

В консорциуме также есть рабочие группы, изучающие зависимости (IGDEP) для обеспечения надежности и устойчивости беспроводной связи, передачи данных с высокой скоростью (HRRCIG) и терагерцовых коммуникаций (THzIG).

Bluetooth

Bluetooth – это технология беспроводной связи с низким энергопотреблением, используемая повсеместно в технологиях от датчиков сотовых телефонов и клавиатур до видеоигр. Название Bluetooth относится к королю Харальду Синезубому (Блатанду) в регионе, где сейчас находится Норвегия и Швеция, примерно в 958 г. Король Блатанд получил свое имя благодаря любви к чернике и/или поеданию замороженных врагов. Несмотря на это, Bluetooth получен от его имени, потому что король Блатанд собрал воюющие племена, и то же самое можно сказать о формировании первоначального Bluetooth SIG. Даже логотип Bluetooth представляет собой комбинацию рун из древнегерманского алфавита, используемого датчанами. Сегодня Bluetooth широко распространен, и этот раздел будет посвящен новому протоколу Bluetooth 5, ратифицированному Bluetooth SIG в 2016 г. Также будут вызваны другие варианты.

В этом разделе будет подробно рассмотрена технология Bluetooth с особым акцентом на новую спецификацию Bluetooth 5.0. Чтобы узнать больше о более старых технологиях Bluetooth, обратитесь к Bluetooth SIG на сайте www.bluetooth.org.

История Bluetooth

Технология Bluetooth была впервые задумана в Ericsson в 1994 г. с намерением заменить неразбериху кабелей и шнуров, соединяющих компьютерную периферию с радиочастотной средой. Intel и Nokia также присоединились к намерениям создать беспроводное соединение сотовых телефонов с компьютерами аналогичным образом. Три из них сформировали SIG в 1996 г. на конференции, проводимой на заводе Ericsson в Лунде, Швеция. К 1998 г. было пять членов Bluetooth SIG: Intel, Nokia, Toshiba, IBM и Ericsson. В этом году была выпущена версия 1.0 спецификации Bluetooth. Версия 2.0 была позже ратифицирована в 2005 г., когда SIG насчитывал более 4000 членов. В 2007 г. Bluetooth SIG работал с Nordic Semiconductor и Nokia для разработки Bluetooth Ultra-Low-Power, который теперь называется **Bluetooth Low Energy (BLE)**. BLE вывела на рынок совершенно новый сегмент устройств, которые могли бы взаимодействовать с использованием «монетных» батарей. К 2010 г. SIG выпустила спецификацию Bluetooth 4.0, которая официально включала BLE. В настоящее время в Bluetooth SIG насчитывается более 2,5 млрд продуктов Bluetooth и 30 000 участников.

Bluetooth широко используется в развертывании IoT в течение некоторого времени, являясь основным устройством при использовании в режиме низкого потребления энергии (LE) для маяков, беспроводных датчиков, систем отслеживания активов, пультов дистанционного управления, мониторов работоспособности и систем сигнализации.

На протяжении всей своей истории Bluetooth и все дополнительные компоненты были под лицензией GPL и, по сути, были с открытым исходным кодом.

История развития технологии Bluetooth, как она развивалась в функциях и свойствах, показана в табл. 5.1.

Таблица 5.1. История развития технологии Bluetooth

Версия	Свойства	Дата выпуска
Bluetooth 1.0 и 1.0B	Bluetooth базового уровня (1 Мб/с). Выпущена первая версия.	1998
Bluetooth 1.1	Стандартизирован IEEE 802.15.1–2002. Убраны дефекты в спецификации 1.0B. Поддержка нешифрованного канала **Received Signal Strength Indicator (RSS**I)	2002
Bluetooth 1.2	IEEE 802.15.1-2005. Быстрое соединение и обнаружение. **Спектр скачкообразного изменения частоты (AFH).** Интерфейс контроллера хоста (трехпроводной UART). Способы управления потоком и повторной передачи	2003

Таблица 5.1 (окончание)

Версия	Свойства	Дата выпуска
Bluetooth 2.0 (+EDR опционально)	**Enhanced Data Rate Mode (EDR)**: 3 Мб/с	2004
Bluetooth 2.1 (+EDR опционально)	**Secure Simple Pairing (SSP):** шифрование публичных ключей с четырьмя уникальными методами аутентификации. **Extended Inquiry Response (EIR)** обеспечивает лучшую фильтрацию и снижение мощности	2007
Bluetooth 3.0 (+ EDR опционально) (+HS опционально)	Режим расширенной ретрансляции L2CAP (ERTM) для надежных и ненадежных состояний соединения. Альтернативный MAC/PHY (AMP) 24 Мбит/с с использованием 802.11 PHY. Одноадресные данные без установления соединения для низкой задержки. Усовершенствованное управление энергопотреблением	2009
Bluetooth 4.0 (+ EDR опционально) (+HS опционально) (+LE опционально)	AKA Bluetooth Smart Представляет режим **Low Energy (LE).** Представлены протоколы и профили ATT и GATT. Двойной режим: режим BR/EDR и LE Менеджер безопасности с AES-шифрованием	2010
Bluetooth 4.1	Сосуществование с **Mobile wireless service (MWS).** Подталкивание (функция сосуществования). Чересстрочное сканирование (функция сосуществования). Устройства поддерживают несколько одновременных ролей	2013
Bluetooth 4.2	Защищенные соединения LE. Конфиденциальность уровня соединения. Профиль поддержки IPv6	2014
Bluetooth 5.0	**Slot availability masks (SAM):** 2 Мбит/с PHY и LE. Режим длинного диапазона LE. Расширенные рекламные режимы LE. Смешанные сети	2016

Топологии и процесс связи Bluetooth 5

Беспроводная связь Bluetooth состоит из двух беспроводных систем: **Basic Rate (BR)** и **Low Energy (LE** или **BLE)**. Узлы могут быть рекламодателями или сканерами по следующему определению:

- **рекламодатель**: устройства, передающие пакеты рекламодателей;
- **сканер**: устройства, принимающие пакеты рекламодателей без намерения подключиться;
- **инициатор** – устройства, пытающиеся сформировать соединение.

В Bluetooth WPAN есть несколько событий Bluetooth:

- **реклама** – инициируется устройством для трансляции на устройства сканирования, чтобы предупредить их о наличии устройства, желающего либо переправить, либо просто передать сообщение в рекламном пакете;
- **подключение** – процесс сопряжения устройства и хоста;
- **периодическая реклама** (для Bluetooth 5) – позволяет рекламному устройству периодически рекламировать 37 непервичных каналов путем перескакивания каналов с интервалом от 7,5 до 81,91875 с;

- **расширенная реклама** (для Bluetooth 5) – позволяет использовать расширенные PDU для поддержки цепочки рекламы и больших PDU, возможно, а также новые варианты использования аудио или других мультимедиа (см. раздел «Маякование» этой главы).

В режиме LE устройство может завершить полное общение, просто используя рекламный канал. В качестве альтернативы связь может требовать двунаправленной канал и принудительно требовать формального подключения устройств. Устройства, которые должны сформировать этот тип соединения, начнут процесс, слушая рекламные пакеты. В этом случае слушатель называется инициатором. Если рекламодатель выдает подключаемое рекламное событие, инициатор может сделать запрос на соединение с использованием того же PHY-канала, на который он получил рекламный пакет.

Рекламодатель может определить, хочет ли он сформировать соединение. Если соединение сформировано, рекламное событие заканчивается, и инициатор теперь называется **мастером**, а рекламодатель называется **ведомым**. Это соединение называется **пикосеть** в Bluetooth-жаргоне, где происходят события по соединению. Все события соединения происходят на одном и том же начальном канале между ведущим и ведомым. После обмена данными и завершения соединения для пары может быть выбран новый канал с использованием скачкообразной перестройки частоты.

Пикосети формируются в двух разных режимах в зависимости от режима BR/EDR или режима BLE. В BR/EDR пикосетка использует трехбитную адресацию и может ссылаться только на семь подчиненных устройств в одной пикосети. Несколько пикосетей могут образовывать объединение, называемое **scatternet**, но здесь должен быть второй мастер для подключения к вторичной сети и управления ею. Ведомый/главный узел берет на себя ответственность за объединение двух пикосетей вместе. В режиме BR/EDR сеть использует один и тот же график скачкообразной перестройки частоты, и все узлы будут гарантированно находиться на одном канале в данный момент времени. В режиме BLE эта система использует 24-битную адресацию, поэтому количество ведомых устройств, связанных с мастером, выражается в миллионах. Каждое отношение «ведущий-ведомый» само по себе является пикосетью и может использовать уникальный канал. В пикосети узлы могут быть **ведущими (M)**, **подчиненными (S)**, **резервными (SB)** или **зарезервированными (P)**. Режим ожидания – это состояние по умолчанию для устройства. В этом состоянии оно имеет возможность находиться в режиме малой мощности потребления. До 255 других устройств могут быть в режиме SB или P в одной пикосети.

Обратите внимание, что Bluetooth 5.0 посчитал устаревшими и удалил зарезервированные состояния в пикосетях; только устройства Bluetooth до версии 4.2 будут поддерживать зарезервированное состояние. Состояние ожидания по-прежнему поддерживается Bluetooth 5.0.

Топология пикосети показана на рис. 5.1.

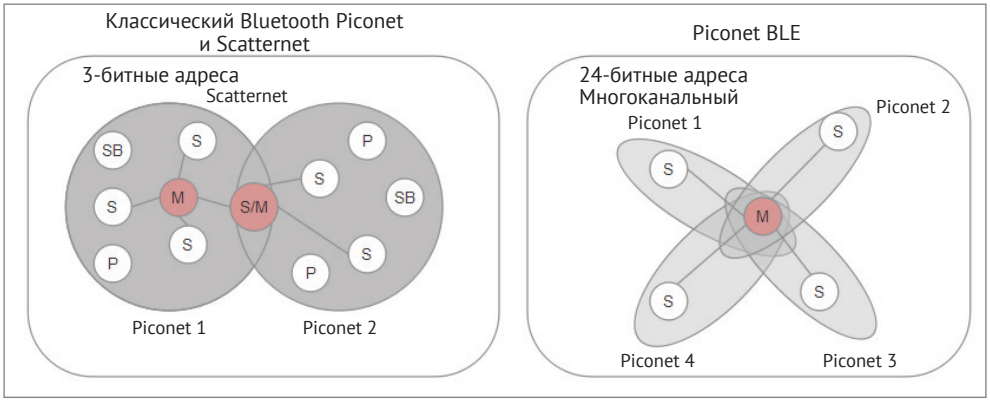


Рис. 5.1 ❖ Разница между классическими (BR/EDR) Bluetooth и BIC-пикосетями. В режиме BR/EDR до семи ведомых устройств можно связать в одну пикосеть из-за 3-битной адресации. Все они имеют общий канал между семью подчиненными. Другие пикосети могут присоединяться к сети и формировать scatternet только в том случае, если присутствует связанный мастер во вторичной сети. В режиме BLE миллионы ведомых устройств могут объединяться в несколько пикосетей с одним мастером из-за 24-разрядной адресации. Каждая пикосеть может быть на другом канале, но только один подчиненный может связываться с мастером в каждой пикосети. Практически говоря, пикосети BLE имеют тенденцию быть намного меньше

Стек Bluetooth 5

Bluetooth имеет три основных компонента: аппаратный контроллер, программное обеспечение для хоста и профили приложений. Устройства Bluetooth поставляются в одно- и двухрежимных версиях, что означает, что они либо поддерживают только стек BLE, либо одновременно поддерживают классический режим и BLE. На рис. 5.1 показано разделение между контроллером и хостом на уровне **интерфейса хост-контроллера (HCI)**. Bluetooth позволяет подключать один или несколько контроллеров к одному хосту.

Стек состоит из уровней или протоколов и профилей:

- **протоколы** – горизонтальные уровни или слои, представляющие функциональные блоки. На рис. 5.2 показан стек протоколов;
- **профили** – представляют вертикальные функции, использующие протоколы. Профили будут подробно описаны в следующем разделе, при этом будут охвачены общие профили атрибутов и общие профили доступа.

На рис. 5.2 представлена всеобъемлющая архитектурная диаграмма стека Bluetooth, включая режимы BR/EDR и BLE, а также режим AMP.

Существуют три режима работы Bluetooth, показанные на рис. 5.2 (каждый из которых требует различные PHY):

- **режим низкого энергопотребления (LE)** – используется полоса ISM 2,4 ГГц и FHSS для защиты от помех. PHY отличается от радиостанций BR/EDR и AMP модуляции, кодирования и скорости передачи данных. LE ра-

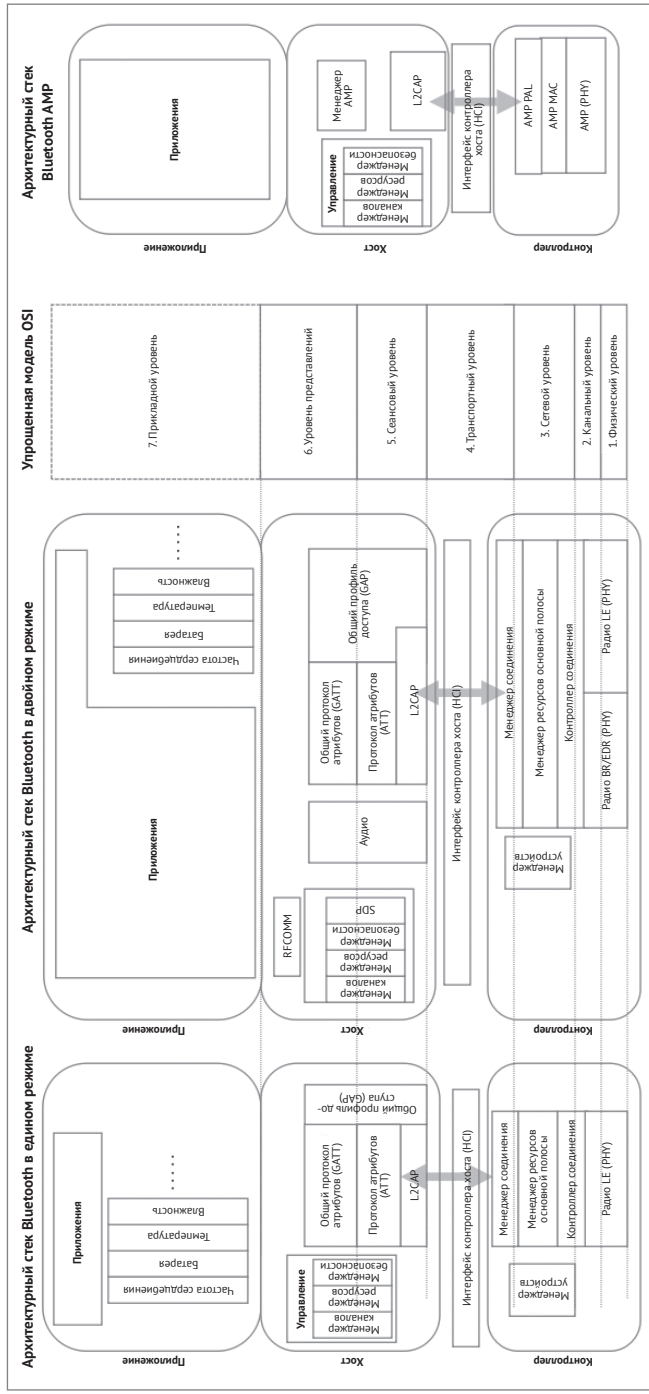


Рис. 5.2 ❖ Bluetooth Single Mode (только BLE) и Dual Mode (Classic и BLE) по сравнению с упрощенным стеком OSI. Правая диаграмма иллюстрирует режим AMP. Обратите внимание на разделение обязанностей между платформой и оборудованием для нижнего стека. HCI – это транспортный канал между оборудованием и хостом

ботает с частотой 1 Мсим/с со скоростью передачи 1 Мбит/с. Bluetooth 5 поддерживает несколько настраиваемых скоростей передачи данных 125 Кбит/с, 500 Кбит/с, 1 Мбит/с и 2 Мбит/с (подробнее об этом позже);

- **режим базовой скорости/улучшенной скорости передачи данных (BR/EDR)** – используется не такое радио, как BLE и AMP, но оно работает в диапазоне ISM 2,4 ГГц. Базовая радиостанция рассчитана на 1 Мсим/с и поддерживает скорость передачи 1 Мбит/с. EDR поддерживает скорость передачи данных 2 или 3 Мбит/с. Это радио использует FHSS для защиты от помех;
- **альтернативный MAC/PHY (AMP)** – это дополнительная функция, которая использует 802.11 для высокоскоростной передачи до 24 Мбит/с. Этот режим требует, чтобы ведущее и ведомое устройство поддерживали AMP. Это вторичный физический контроллер, но он требует, чтобы система имела контроллер BR/EDR для установления начального соединения и согласования.

Теперь мы подробно рассмотрим функцию каждого элемента стека. Мы начинаем с общих блоков BR/EDR и LE, а затем перечисляем детали для AMP. Во всех трех случаях мы начнем с физического уровня и переместимся по стеку к прикладному уровню.

Основные архитектурные блоки:

- уровень контроллера:
  - **BR/EDRPHY (блок контроллера)** – отвечает за передачу и прием пакетов по физическому каналу на 79 каналах;
  - **LEPHY** – физический интерфейс с низкой энергией, ответственный за управление 40 каналами и скачкообразной перестройкой частоты;
  - **контроллер соединения** – кодирует и декодирует пакеты Bluetooth из полезной нагрузки данных;
  - **менеджмент ресурсов базовой полосы** – отвечает за весь доступ к радио из любого источника. Управляет планированием физических каналов и согласовывает контракты доступа со всеми объектами для обеспечения соответствия параметров QoS;
  - **менеджер соединений** – создает, изменяет и освобождает логические соединения и параметры обновления, связанные с физическими связями между устройствами. Повторно используется для режимов BR/EDR и LE и разных протоколов;
  - **диспетчер устройств** – блокирует уровень основной полосы контроллера, который контролирует общее поведение Bluetooth. Отвечает за все операции, не связанные с передачей данных, включая создание устройств, которые можно обнаружить или подключить, подключиться к устройствам и сканировать устройства;
  - **интерфейс хост-контроллера (HCI)** – это разделение между хостом и физическим контроллером на четвертом уровне сетевого стека. Он предоставляет интерфейсы, позволяющие хосту добавлять, удалять, управлять и обнаруживать устройства в пикосети;

- уровень хоста:
  - **L2CAP** – это логический протокол управления связью и адаптации. Он используется для мультиплексирования логических соединений между двумя различными устройствами с использованием протоколов более высокого уровня, чем физический. Он может сегментировать и собирать пакеты;
  - **менеджер каналов** – отвечает за создание, управление и закрытие каналов L2CAP. Мастер будет использовать протокол L2CAP для связи с менеджером подчиненного канала;
  - **менеджер ресурсов** – отвечает за порядок подачи фрагментов на уровень основной полосы. Помогает обеспечить качество обслуживания;
  - **протокол диспетчера безопасности (SMP)** – также известен как протокол диспетчера безопасности. Этот блок отвечает за генерацию ключей, квалификационных ключей и сохранение ключей;
  - **протокол обнаружения служб (SDP)** – обнаружение услуг, предлагаемых на других устройствах с помощью UUID;
  - **аудио** – дополнительный профиль эффективного воспроизведения потокового аудио;
  - **RFCOMM** – этот блок отвечает за эмуляцию и интерфейс RS-232 и используется для поддержки функциональности телефонии;
  - **протокол атрибута (ATT)** – протокол проводного приложения, используемый главным образом в BLE (но может применяться в BR/EDR). Оптимизирован для работы на оборудовании на базе батарей BLE. ATT тесно связан с GATT;
  - **главный профиль атрибута (GATT)** – этот блок представляет функциональность сервера атрибутов и, необязательно, клиента атрибута. Профиль описывает службы, используемые в сервере атрибутов. Каждое устройство BLE должно иметь профиль GATT. В принципе, если не исключительно, используется для BLE, но может использоваться на простаивающих устройствах BR/EDR;
  - **общий профиль доступа (GAP)** – контролирует соединения и состояния рекламирования. Позволяет устройству быть видимым для внешнего мира и составляет основу всех других профилей;
- стек, специфичный для AMP:
  - **AMP (PHY)** – физический уровень, ответственный за передачу и прием пакетов данных до 24 Мбит/с;
  - **AMP MAC** – это уровень управления доступом к физической среде, как определено в модели IEEE 802. Он предоставляет методы адресации устройств;
  - **AMP PAL** – уровень, соединяющий AMP MAC с хост-системой (L2CAP и AMP-менеджер). Этот блок преобразует команды с хоста в конкретные примитивы MAC и наоборот;

– **менеджер AMP** – использует L2CAP для связи с одноранговым менеджером AMP на удаленном устройстве. Обнаруживает удаленные устройства AMP и определяет их доступность.

PHY Bluetooth 5 и интерференция

Устройства Bluetooth работают в промышленном, научном и медицинском (ISM) нелицензионном диапазоне частот от 2,4000 до 2,4835 ГГц. Как упоминалось ранее в этой главе, эта конкретная нелицензированная область переполнена рядом других беспроводных носителей, таких как 802.11 Wi-Fi. Чтобы уменьшить помехи, Bluetooth поддерживает **широкополосный спектр с перестройкой частоты (FHSS)**.

Когда у вас есть выбор между классическими режимами Bluetooth BR/EDR, EDR будет иметь меньше шансов для интерференции и лучшее сосуществование с Wi-Fi и другими устройствами Bluetooth, поскольку время нахождения в эфире меньше из-за скорости.

Адаптивная перестройка частоты (AFH) была введена в Bluetooth 1.2. AFH использует два типа каналов: используемые и неиспользуемые. Используемые каналы являются задействованными как часть последовательности прыжков. Неиспользуемые каналы заменяются в последовательности скачкообразной перестройки используемыми каналами, когда это необходимо в методе случайной замены. Режим BR/EDR имеет 79 каналов, а BLE – 40 каналов. При использовании 79 каналов режим BR/EDR имеет менее 1,5% вероятности интерференции с другим каналом. Это позволяет офисной установке иметь сотни наушников, периферийных устройств и устройств, находящихся в одном и том же диапазоне, в зависимости от частотного пространства (например, фиксированного и непрерывного использования источников интерференции).

AFH позволяет подчиненному устройству сообщать информацию о классификации канала ведущему устройству, чтобы помочь в настройке переключения каналов. В ситуациях, когда есть интерференция с 802.11 Wi-Fi, AFH используется с комбинацией проприетарных методов для определения приоритетности трафика между двумя сетями. Например, если последовательность перескоков регулярно сталкивается на канале 11, мастер и ведомые устройства в пикосети просто договорятся переходить через канал 11 в будущем.

В режиме BR/EDR физический канал делится на слоты. Данные позиционируются для передачи в точном слоте, и при необходимости могут использоваться последовательные слоты. Используя этот метод, Bluetooth обеспечивает эффект полнодуплексной связи посредством **дуплекса с временным разделением (TDD)**. BR использует **гауссову модуляцию перестройку частоты** (**GFSK**) для достижения скорости 1 Мбит/с, в то время как EDR использует **модуляцию дифференциальной четвертичной фазовой манипуляции (DQPSK)** до 2 Мбит/с и 8-фазную дифференциальную фазовую синхронизацию (8DPSK) со скоростью 3 Мбит/с.

LE, с другой стороны, использует схемы с **множественным доступом с частотным разделением каналов (FDMA)** и **множественным доступом**

с временным разделением (TDMA). При использовании 40 каналов вместо 79 для BR/EDR и каждого канала, разделенного на 2 МГц, система будет делить 40 каналов на три для рекламы и 37 – на вторичную рекламу и данные. Каналы Bluetooth выбираются псевдослучайно и переключаются со скоростью 1600 прыжков в секунду. На рис. 5.3 показано распределение и разбиение частоты BLE в пространстве ISM 2,4 ГГц.

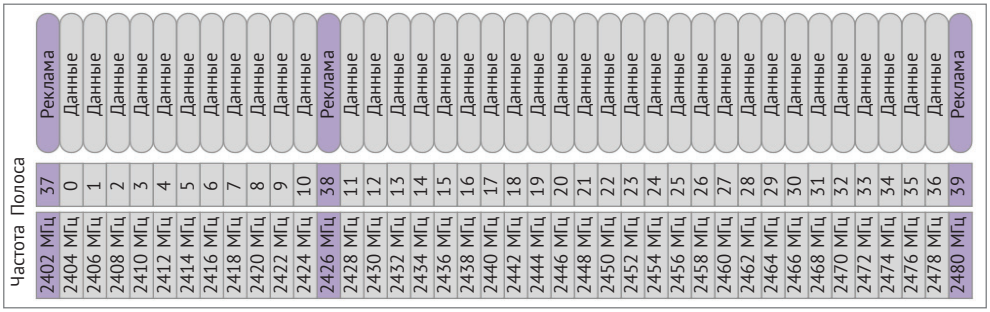


Рис. 5.3 ❖ Частоты BLE разделены на 40 уникальных полос с разделением 2 МГц. 3 канала, предназначенные для рекламы и остальные 37, предназначенные для передачи данных

TDMA используется для организации связи, требуя, чтобы одно устройство передавало пакет в заданное время и принимающее устройство отвечало в другое заданное время. Физические каналы подразделяются на единицы времени для определенных событий LE, таких как реклама, периодическая реклама, расширенная реклама и подключение. В LE мастер может сформировать связь между несколькими подчиненными устройствами. Аналогичным образом, ведомый может иметь несколько физических связей с несколькими ведущими устройствами, а устройство может одновременно быть как ведущим, так и подчиненным. Изменения роли от ведущего к ведомому или наоборот не допускаются.

Как отмечалось ранее, 37 из 40 каналов предназначены для передачи данных, но три предназначены для рекламы. Каналы 37, 38 и 39 посвящены рекламным профилям GATT. Во время рекламы устройство будет передавать рекламный пакет по всем трем каналам одновременно. Это помогает увеличить вероятность того, что сканирующее хост-устройство увидит рекламу и ответит.

Другие формы интерференции возникают с мобильными стандартами беспроводной связи в пространстве 2,4 ГГц. Здесь в Bluetooth 4.1 был введен метод подталкивания поезда.

Bluetooth 5.0 представил **маски доступности слотов (SAM)**. SAM позволяет двум устройствам Bluetooth указывать друг другу временные слоты, доступные для передачи и приема. Создана карта с указанием доступности временных слотов. Благодаря отображе-

нию, контроллеры Bluetooth могут уточнить свои временные слоты BR/EDR и повысить общую производительность.

Для режимов BLE SAM недоступен. Однако пересмотренный механизм в Bluetooth, называемый **алгоритмом выбора канала 2 (CSA2)**, может помочь в скачкообразной перестройке частоты в шумных средах, восприимчивых к замиранию многолучевости. CSA2 был введен в Bluetooth 4.1 и является очень сложным алгоритмом отображения каналов и прыжков. Он улучшает устойчивость радио к интерференции и позволяет радио ограничить количество радиочастотных каналов, которые могут использоваться в местах с высокой интерференцией. Побочным эффектом ограничения каналов с CSA2 является то, что мощность передачи увеличивается до +20 дБм. Как уже упоминалось, существуют ограничения на мощность передачи, налагаемые регулирующими органами, поскольку рекламных каналов BLE и подключенных каналов так мало. CSA2 позволяет больше доступных каналов в Bluetooth 5, чем в предыдущих версиях, что поможет преодолеть регуляторные ограничения.

Структура пакета Bluetooth

Каждое устройство Bluetooth имеет уникальный 48-битный адрес, называемый `BD_ADDR`. Верхние 24 бита `BD_ADDR` относятся к конкретному адресу производителя и приобретаются через Центр регистрации IEEE. Этот адрес включает **уникальный идентификатор организации (OUI)**, также известный как идентификатор компании, и назначается IEEE. 24 наименее значимых бита для компании могут быть изменены.

Три других безопасных и выбранных случайным образом формата адреса доступны, но будут обсуждаться в разделе безопасности BLE этой главы. На рис. 5.4 показана структура рекламных пакетов BLE и различные типы PDU. Представлены некоторые из наиболее часто используемых PDU.

Работа BR/EDR

Режим классического Bluetooth (BR/EDR) ориентирован на соединение. Если устройство подключено, связь поддерживается, даже если данные не передаются. Прежде чем какое-либо соединение Bluetooth появится, устройство должно быть обнаружено, чтобы оно отвечало на сканирование физического канала и впоследствии отвечало адресом своего устройства и другими параметрами. Устройство должно находиться в режиме подключения для контроля сканирования страницы.

Процесс подключения выполняется в три этапа:

1) **запрос** – на этом этапе два устройства Bluetooth никак не ассоциированы или не *связаны*; они ничего не знают друг о друге. Устройства должны обнаружить друг друга через запрос. Если другое устройство слушает, оно может ответить своим адресом `BR_ADDR`;
2) **пейджинг** – это образование соединения между двумя устройствами. Каждое устройство знает `BD_ADDR` друг от друга в этот момент;
3) **подключено** – существует четыре подрежима состояния подключения. Это нормальное состояние, когда два устройства активно общаются:

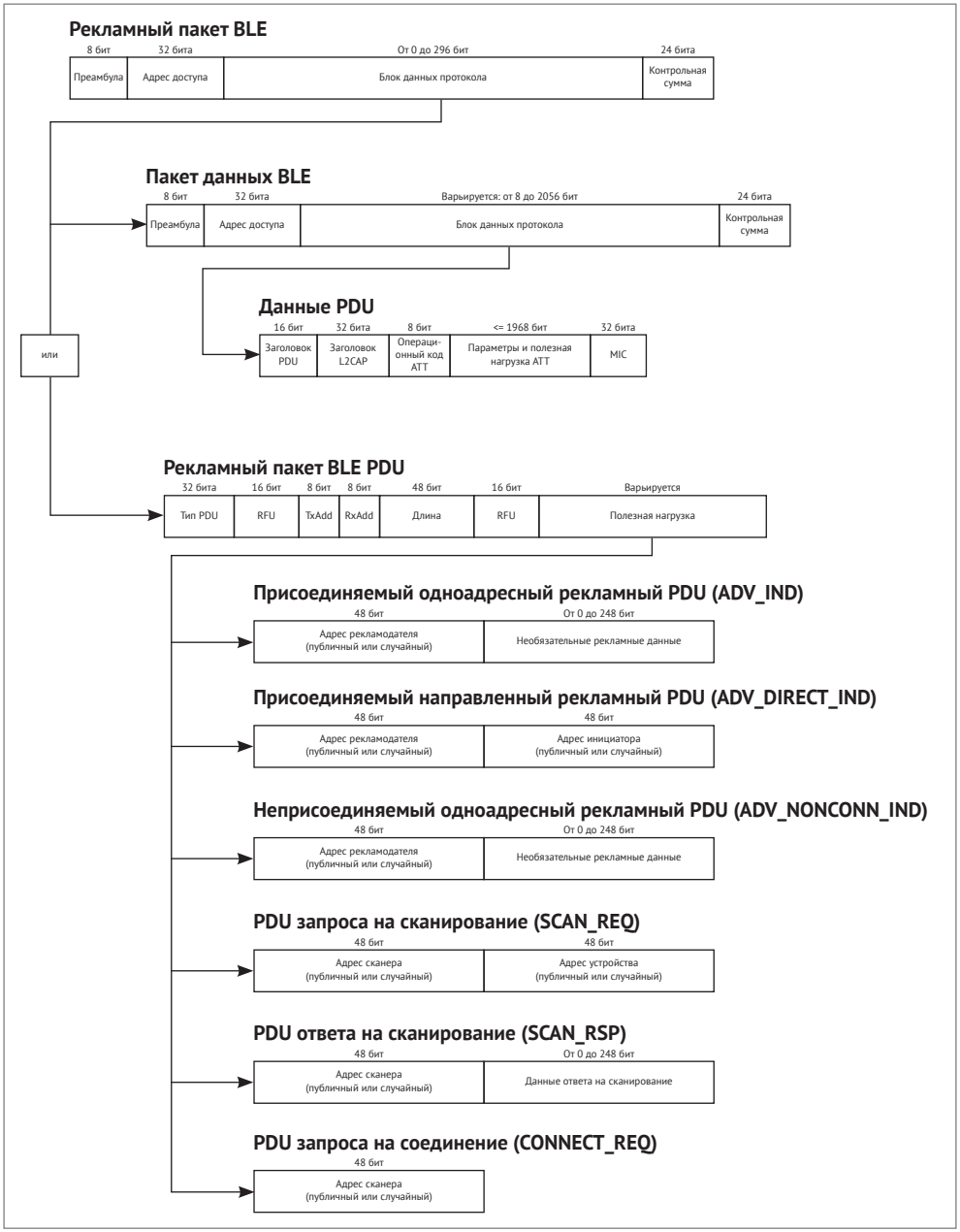


Рис. 5.4 ❖ Обычные форматы рекламы BLE и пакетов данных. Существует несколько других типов пакетов, на которые необходимо ссылаться в спецификации Bluetooth 5.0

а) **активный режим** – это нормальный режим работы для передачи и приема данных Bluetooth или ожидания следующего слота передачи;
б) **режим анализа** – режим энергосбережения. Устройство, по существу, спит, но будет прослушивать передачи во время определенных слотов, которые могут быть изменены программно (например, 50 мс);
в) **режим ожидания** – это временный режим низкого энергопотребления, инициированный ведущим или ведомым устройством. Устройство не будет слушать передачи, как в режиме анализа, и ведомый временно игнорирует ACL-пакеты. В этом режиме переключение на подключенное состояние происходит очень быстро;
г) **режим парковки** – как указано ранее, этот режим устарел в Bluetooth 5.

Диаграмма состояния этих фаз выглядит следующим образом (рис. 5.5).

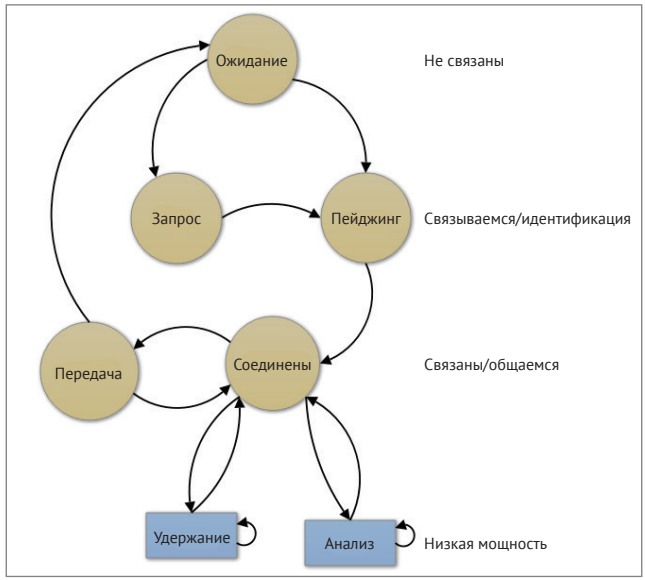


Рис. 5.5 ❖ Процесс подключения для Bluetooth от неподключенного устройства в режиме ожидания, запрос устройства и обнаружение, режим подключения/передачи и режимы с низким энергопотреблением

Apple рекомендует использовать режим анализа с интервалом 15 мс. Это экономит значительную мощность за счет сохранения устройства в активном режиме, но также позволяет лучше использовать спектр с помощью Wi-Fi и других сигналов Bluetooth в этой области. Кроме того, Apple рекомендует устройству сначала установить рекламный интервал для первоначального обнаружения хостом до 20 мс и транслировать его в течение 30 с. Если устройство по-прежнему не может подключиться к хосту, интервал рекламы должен быть увеличен программно, чтобы увеличить вероятность завершения

процесса подключения. См. Руководство по разработке аксессуаров Bluetooth для продуктов Apple Release 8, Apple Computer, 16 июня 2017 г.

Если этот процесс завершится успешно, два устройства могут быть принудительно автоматически подключены, когда они находятся в зоне действия. Теперь устройства сопрягаются. Одноразовый процесс сопряжения наиболее распространен в подключении смартфонов к стереосистеме автомобиля, но он может применяться в любом месте и в IoT. Сопряженные устройства будут использовать ключ, который используется в процессе аутентификации. Более подробно будут рассмотрены ключи и аутентификация в разделе безопасности Bluetooth.

Работа BLE

В режиме низкой энергии Bluetooth существует пять состояний связи, которые согласованы хостом и устройством:

- **рекламирование** – устройства, которые передают рекламные пакеты по рекламным каналам;
- **сканирование** – устройства, которые получают рекламу по рекламным каналам без намерения подключиться. Сканирование может быть активным или пассивным:
  - **активное сканирование** – канальный уровень прослушивается на предмет рекламных блоков PDU. В зависимости от принятого PDU он может потребовать от рекламодателя отправить дополнительную информацию;
  - **пассивное сканирование** – канальный уровень будет только принимать пакеты, передача отключена;
- **инициирование** – устройства, которым необходимо сформировать соединение с другим устройством, прослушивают подключаемые рекламные пакеты и инициируют отправку пакета подключения;
- **подключено** – между подключенным ведущим и ведомым есть связь. Ведущий – инициатор, а ведомый – рекламодатель:
  - **центральный** – инициатор преобразует роль и заголовок в центральное устройство;
  - **периферийный** – рекламирующее устройство становится периферийным устройством;
- **режим ожидания** – устройство в неподключенном состоянии.

Состояние рекламы имеет несколько функций и свойств. Объявления могут быть общей рекламой, когда устройство транслирует общее приглашение на другое устройство в сети. Ориентированная реклама уникальна и предназначена для того, чтобы пригласить конкретного партнера для подключения как можно быстрее. Этот режим рекламы содержит адрес рекламного устройства и приглашенного устройства.

Когда принимающее устройство распознает пакет, оно немедленно отправит запрос на соединение. Направленная реклама – это быстрое и немедленное внимание, а рекламные объявления отправляются со скоростью 3,75 мс, но только на 1,28 с. Неподсоединяемая реклама, по существу, является маяком (и может

даже не нуждаться в приемнике). Ниже мы опишем маяки. Наконец, обнаруживаемая реклама может отвечать на запрос сканирования, но не будет принимать соединения. На рис. 5.6 состояний показаны пять состояний связи операции BLE.

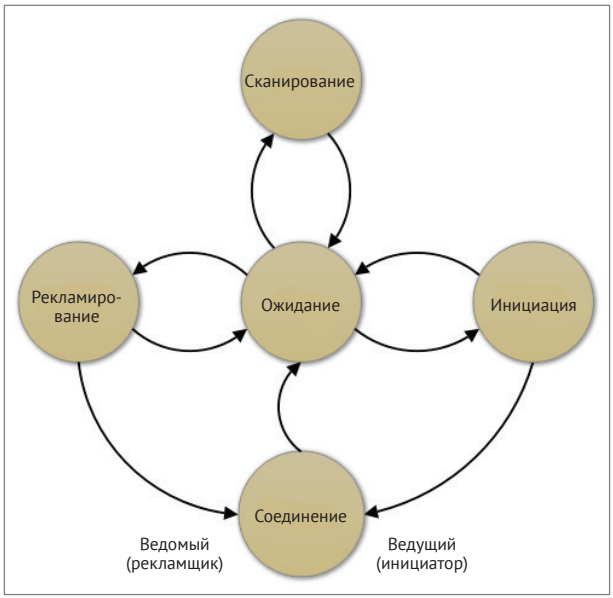


Рис. 5.6 ❖ Состояния связи BLE

Устройство BLE, которое ранее не связывалось с хостом, инициирует коммуникацию, транслируя рекламные объявления по трем рекламным каналам. Хост может ответить `SCAN_REQ`, чтобы запросить дополнительную информацию с рекламного устройства. Периферийное устройство отвечает `SCAN_RSP` и включает имя устройства или, возможно, службы.

`SCAN_RSP` может повлиять на использование энергии на периферийном устройстве. Если устройство поддерживает ответы на сканирование, оно должно поддерживать свое радио активным в режиме приема, потребляя энергию. Это происходит, даже если ни одно хост-устройство не выдает `SCAN_REQ`. Целесообразно отключить ответы сканирования на периферийных устройствах IoT, имеющими ограничения по питанию.

После сканирования хост (сканер) инициирует `CONNECT_REQ`, после чего сканер и рекламодатель отправят пустые пакеты PDU для указания подтверждения. Сканер теперь называется мастером, а рекламодатель называется ведомым. Мастер может обнаруживать профили ведомых и службы через GATT. После того, как обнаружение завершено, данные могут обмениваться от подчиненного устройства к мастеру и наоборот. По завершении мастер вернется в режим сканирования, и ведомое устройство вернется в режим рекламодателя. Рисунок 5.7 иллюстрирует процесс сопряжения BLE от рекламы до передачи данных.

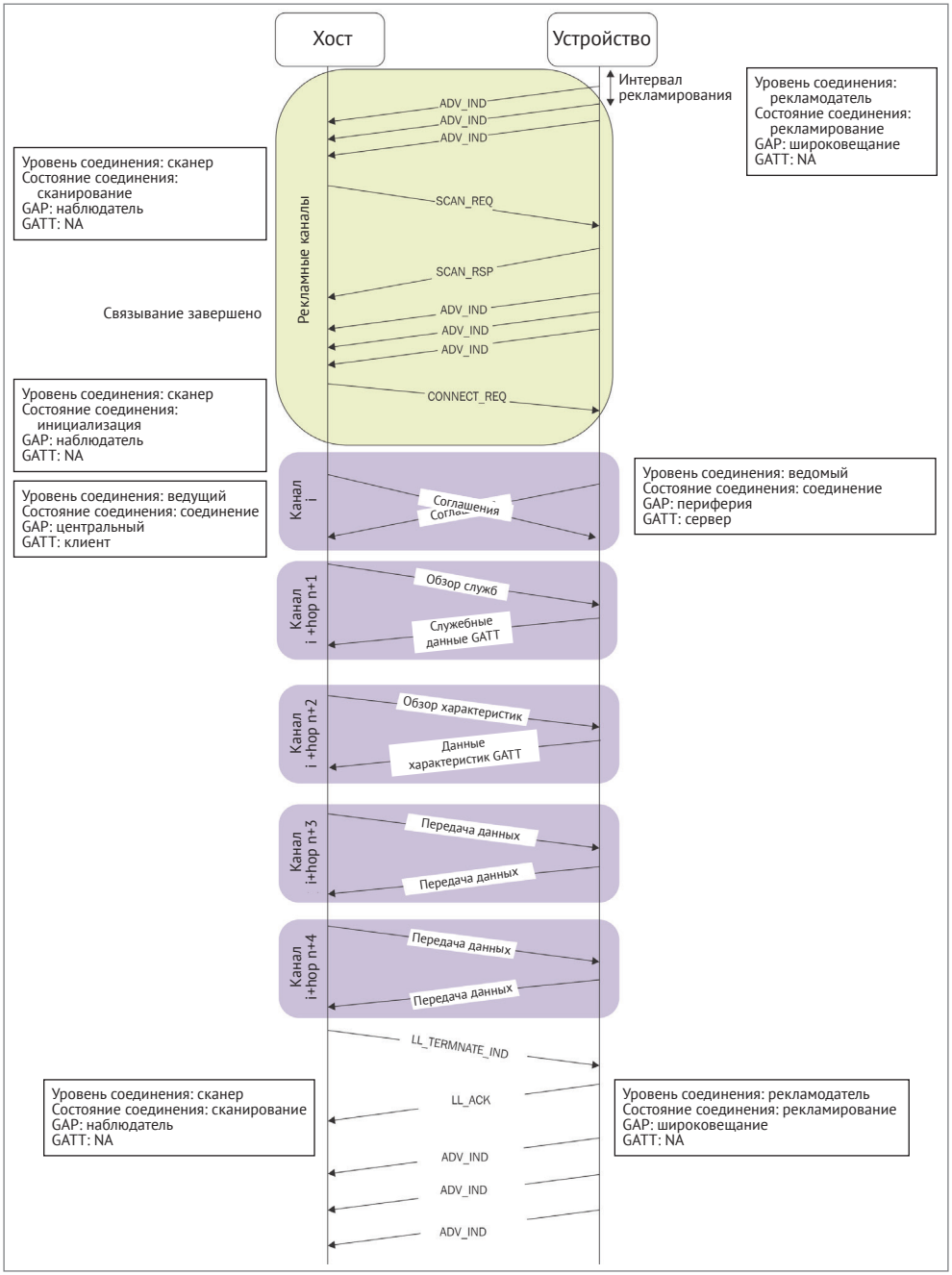


Рис. 5.7 ❖ Фазы рекламы BLE, подключения, запроса услуг GATT и передачи данных

Профили Bluetooth

Интерфейс приложений с различными устройствами Bluetooth с использованием профилей. Профили определяют функциональные возможности и функции для каждого уровня стека Bluetooth. По сути, профили связывают стек вместе и определяют, как уровни взаимодействуют друг с другом. Профиль описывает характеристики обнаружения, которые рекламирует устройство. Они также используются для описания форматов данных услуг и характеристик, которые приложения используют для чтения и записи на устройства. Профиль не существует на устройстве; скорее, они являются предопределенными конструкциями, поддерживаемыми и управляемыми Bluetooth SIG.

Основной профиль Bluetooth должен содержать GAP, как указано в спецификации. GAP определяет радио, уровень основной полосы, диспетчер связей, L2CAP и обнаружение службы для устройств BR/EDR. Аналогично, для устройства BLEGAP будет определять радио, уровень канала, L2CAP, диспетчер безопасности, протокол атрибутов и общий профиль атрибутов.

Протокол атрибутов ATT представляет собой проводной протокол клиент-сервер, оптимизированный для маломощных устройств (например, длина никогда не передается через BLE, это подразумевается размером PDU). ATT также очень общий, и многое остается для GATT для оказания помощи. Профиль ATT состоит из:

- 16-разрядного дескриптора;
- UUID для определения типа атрибута;
- значения, содержащего длину.

GATT логически находится поверх ATT и используется в основном, если не исключительно, для устройств BLE. GATT определяет роли сервера и клиента. Клиент GATT обычно является периферийным устройством, а сервером GATT является хост (ПК, смартфон). Профиль GATT содержит два компонента:

- **службы** – служба разбивает данные на логические объекты. В профиле может быть несколько служб, и каждая служба имеет уникальный UUID, чтобы отличать ее от других;
- **характеристики** – характеристикой является самый низкий уровень профиля GATT, он содержит необработанные данные, связанные с устройством. Форматы данных отличаются 16-разрядными или 128-битными UUID. Дизайнер может свободно создавать свои собственные характеристики, которые может интерпретировать только их приложение.

На рис. 5.8 представлен пример профиля BluetoothGATT с соответствующими UUID для различных сервисов и характеристик.

Bluetooth SIG поддерживает коллекцию многих профилей GATT. На момент написания документа на Bluetooth SIG поддерживалось 57 профилей GATT: www.bluetooth.com/specifications/gatt. Профили, поддерживаемые SIG, включают в себя все: от мониторов здоровья, велосипедных и фитнес-устройств до мониторов окружающей среды, устройств интерфейса пользователя, внутрен-

него позиционирования, передачи объектов и местоположения и навигационных служб, а также многие другие.

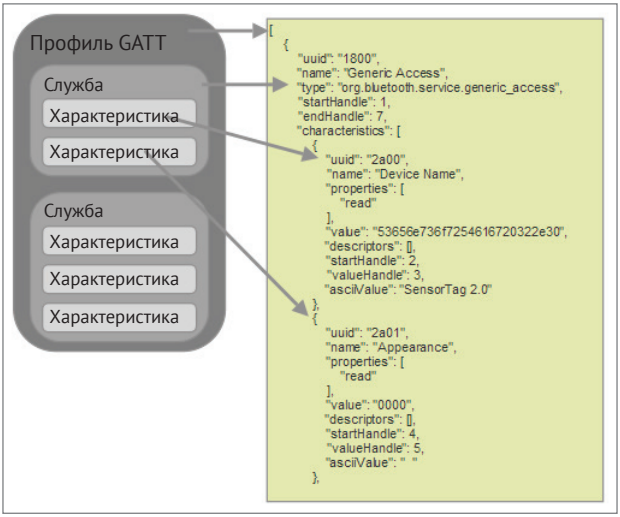


Рис. 5.8 ❖ Иерархия профиля GATT и пример GATT, используемый в Texas Instruments CC2650 SensorTag

Безопасность BR/EDR

Безопасность Bluetooth существует как часть протокола с 1.0 в той или иной форме. Мы обсуждаем безопасность режима BR/EDR и BLE отдельно, поскольку механизмы разные. Начиная с режима BR/EDR, существует несколько режимов аутентификации и сопряжения. Для обеспечения безопасности BR/EDR и BLE рекомендуется прочитать и следовать последнему руководству по безопасности, предоставленному Национальным институтом стандартов и технологий США: *Руководство по безопасности Bluetooth, Специальная публикация NIST* (*SP*), *800-121 Rev. 2, NIST , 5/8/2017*.

Для сопряжения требуется генерация секретного симметричного ключа. В режиме BR/EDR это называется ключом соединения, тогда как в режиме BLE он называется *долгосрочным ключом*. Старые устройства Bluetooth использовали режим сопряжения **персонального идентификационного номера (PIN)**, чтобы инициировать ключи соединения. Новые устройства (4.1+) используют безопасное простое сопряжение.

Безопасное простое сопряжение (SSP) обеспечивает процесс сопряжения с помощью ряда различных моделей ассоциаций для различных случаев использования. SSP также использует криптографию с открытым ключом для защиты от подслушивания и атак типа **«человек-посредине» (MITM)**. Модели, поддерживаемые SSP, включают:

- **числовое сравнение** – для случаев использования, когда оба устройства Bluetooth могут отображать шестизначное числовое значение, позволяющее пользователю вводить ответ «да»/«нет» на каждом устройстве, если числа совпадают;
- **ввод общего ключа** – используется в ситуациях, когда одно устройство имеет числовой дисплей, а другое имеет цифровую клавиатуру. В этом случае пользователь вводит значение, отображаемое на дисплее первого устройства на клавиатуре второго устройства;
- **JustWorks™** – для ситуаций, когда одно устройство не имеет клавиатуры или дисплея. Он обеспечивает минимальную аутентификацию и не будет препятствовать атаке MITM;
- **внеполосный (OOB)** – используется, когда устройство имеет дополнительную форму связи, например, NFC или Wi-Fi. Вторичный канал используется для обнаружения и обмена криптографическими значениями. Он защитит только от подслушивания и MITM, если канал OOB защищен.

Аутентификация в режиме BR/EDR – это вызов-ответ; например, ввод PIN-кода на клавиатуре. Если аутентификация не удалась, устройство будет ожидать некоторое время, прежде чем разрешить новую попытку. Интервал растет экспоненциально с каждой неудачной попыткой. Это просто разочаровывает человека, пытающегося вручную подобрать код ключа.

Шифрование в режиме BR/EDR может быть установлено таким образом, чтобы оно было отключено для всего трафика, так что зашифрован трафик данных, но широковещательная связь будет необработанной или чтобы вся связь была зашифрована. Шифрование использует криптографию AES-CCM.

Безопасность BLE

Сопряжение BLE (объяснено ранее в этой главе) начинается с устройства, инициирующего *Pairing_Request* и обмениваясь способностями, требованиями и т. д. На начальном этапе процесса сопряжения не возникает ничего, связанное с профилями безопасности. В этом отношении безопасность сопряжения аналогична четырем методам BR/EDR (также известным как модели ассоциации), но в Bluetooth BLE 4.2 немного отличается:

- **числовое сравнение** – это то же самое, что и JustWorks, но в конце оба устройства генерируют значение подтверждения, которое отображается на экранах хоста и устройства, чтобы пользователь мог проверить соответствие;
- **ввод общего ключа** – подобно режиму BR/EDR, за исключением того, что неинициализирующее устройство создает случайное 128-битное начальное число, которое называется случайным кодом для аутентификации соединения. Каждый бит ключа доступа аутентифицируется отдельно на каждом устройстве, генерируя значение подтверждения для этого бита. Значения подтверждения обмениваются и должны совпадать. Про-

цесс продолжается до тех пор, пока все биты не будут обработаны. Это обеспечивает довольно надежное решение для атак MITM;

- **JustWorks™** – после того, как устройства обмениваются открытыми ключами, неинициализирующее устройство создает случайный код для создания значения подтверждения Cb. Он передает случайный код и Cb на инициирующее устройство, которое, в свою очередь, генерирует свой собственный случайный код и передает его первому. Инициирующее устройство затем подтвердит подлинность неинициализирующего случайного кода, создав собственное значение Ca, которое должно соответствовать Cb. Если оно не соответствует, соединение повреждено. Это опять-таки отличается от режима BR/EDR;
- **внеполосный (OOB)** – это то же самое, что и в режиме BR/EDR. Он защитит только от подслушивания и MITM, если канал OOB защищен.

В BLE (как в случае Bluetooth 4.2) генерация ключей использует безопасные соединения LE. Безопасное соединение LE было разработано для решения проблемы безопасности в сопряжении BLE, что позволило прослушивателю увидеть обмен сопряжения. В этом процессе для шифрования соединения используется **долгосрочный ключ (LTK)**. Ключ основан на криптографии с открытым ключом **алгоритмом Диффи-Хеллмана на эллиптических кривых (ECDH)**. И ведущий, и ведомый будут генерировать пары ключей публичный – частный ECDH. Эти два устройства будут обмениваться публичными частями своих соответствующих пар и обрабатывать ключ Диффи-Хеллмана. На этом этапе соединение может быть зашифровано с использованием криптографии AES-CCM.

BLE также имеет возможность сделать случайным свой `BD_ADDR`. Помните, что `BD_ADDR` – это 48-битный MAC-адрес. Вместо статического адреса для значения, как упоминалось ранее в этой главе, есть еще три варианта:

- **случайный статический** – эти адреса либо прожигаются на устройстве во время изготовления, либо генерируются при циклическом включении питания устройства. Если устройство обычно работает циклически, генерируется уникальный адрес, и это остается защищенным, пока частота циклов питания высока. В среде датчика IoT это может быть не так;
- **случайный частный разрешимый** – этот метод адресации может использоваться только в том случае, если в течение процесса связывания обменяться идентификационным ключом (IRK) между двумя устройствами. Устройство будет использовать IRK для кодирования своего адреса в случайный адрес в рекламном пакете. Второе устройство, которое также имеет IRK, преобразует случайный адрес обратно в аутентичный адрес. В этом методе устройство будет периодически генерировать новый случайный адрес, основанный на IRK;
- **случайный частный неразрешимый** – адрес устройства представляет собой просто случайное число, и в любой момент может быть сгенерирован новый адрес устройства. Это обеспечивает самый высокий уровень безопасности.

Маякование

Маякование Bluetooth – вторичный эффект BLE; однако это важная и значимая технология для IoT. Поскольку маяки не обязательно являются датчиками, мы не рассматривали их явно в главе 3 (хотя некоторые из них предоставляют информацию об обнаружении в рекламном пакете). Маяк просто использует устройства Bluetooth в режиме LE для рекламы на периодической основе. Маяки никогда не соединяются и не сопрягаются с хостом. Если бы маяк подключался, все рекламные объявления останавливались, и никакое другое устройство не могло слышать этот маяк. Три случая использования, важные для розничной торговли, здравоохранения, отслеживания активов, логистики и многих других рынков:

- статическая точка интереса (POI);
- распространение телеметрических данных;
- позиционирование в помещении и геолокация.

В рекламе Bluetooth используется сообщение, которое содержит дополнительную информацию в широковещательном UUID. Приложение на мобильном устройстве может отреагировать на это объявление и выполнить какое-либо действие, если получено правильное объявление. В типичном случае для розницы будет использоваться мобильное приложение, которое будет реагировать на присутствие рекламы маяка поблизости и отображать информацию о рекламе или продаже на мобильном устройстве пользователя. Мобильное устройство будет связываться через Wi-Fi или сотовую связь, чтобы получить дополнительный контент, а также предоставить важные данные о рынке и покупателе для компании.

Маяки могут передавать свою калиброванную мощность сигнала RSSI в качестве рекламы. Сила сигнала маяков калибруется производителями, как правило, с расстоянием в один метр. Внутренняя навигация может выполняться тремя способами:

- **несколько маяков в комнате** – это простой метод триангуляции для определения местоположения пользователя на основе объявленной мощности сигнала RSSI, собранной от многочисленных маяков в комнате. Учитывая рекламируемый калиброванный уровень от каждого маяка и полученную силу от каждого маяка, алгоритм может определить приблизительное местоположение приемника в комнате. Это предполагает, что все маяки находятся на фиксированных местах;
- **один маяк в комнате** – по этой схеме в каждую комнату помещается один маяк, позволяющий пользователю перемещаться между комнатами и залами с точностью до места, где находится комната. Это полезно в музеях, аэропортах, концертных залах и т. д.;
- **несколько маяков на одно здание** – в сочетании с акселерометрами и гироскопами в мобильном устройстве несколько маяков в здании могут допускать способность к ориентации в большом открытом пространстве. Это позволяет одному маяку устанавливать начальное местополо-

жение, а мобильному устройству – определять его местоположение на основе движения пользователя.

Существуют два основных протокола радиомаяка: Eddystone от Google и iBeacon от Apple. Устаревшие устройства Bluetooth могут поддерживать только маяковое сообщение длиной 31 байт. Это ограничивает объем данных, которые может передать устройство. Разработанные схемы были использованы для уменьшения размера пакета и кодирования сообщения.

Полное сообщение iBeacon – это просто UUID (16 байт), большое число (два байта) и младшее число (два байта). UUID специфичен для приложения и использования. Основное число дополнительно уточняет случай использования, а второстепенное расширяет прецедент до более узкого случая.

iBeacons предоставляют два способа обнаружения устройств:

- **мониторинг** – это работает, даже если соответствующее приложение для смартфонов активно не работает;
- **дальнометрия** – измерение дальности работает только в том случае, если приложение активно.

Eddystone (также известный как UriBeacons) может передавать четыре разных типа кадров с различной длиной и кодировкой кадра:

- **Eddystone-URL** – это единое расположение ресурсов. Этот кадр позволяет приемному устройству отображать веб-контент на основе местоположения маякового радиосигнала. Чтобы активировать контент, приложение не нужно устанавливать. Содержимое имеет переменную длину и применяет уникальные схемы сжатия, чтобы уменьшить размер URL-адреса до предела в 17 байт;
- **Eddystone-UID** – уникальный 16-байтовый идентификатор маяка с 10-байтовым пространством имен и шестибайтовым экземпляром. Использует реестр маяка Google для возврата вложений;
- **Eddystone-EID** – короткоживущий идентификатор для маяков, требующих более высокого уровня безопасности. Нет фиксированного пространства имен и идентификатора, идентификаторы постоянно вращаются и требуют разрешенного приложения для декодирования. Использует реестр маяка Google для возврата вложений;
- **Eddystone-TLM** – транслирует телеметрические данные о самом маяке (уровень заряда батареи, время с момента включения, количество рекламы). Широковещательные передачи наряду с URI или URL-пакетами.

Рисунок 5.9 иллюстрирует структуру пакета рекламы Bluetooth BLE для Eddystone и iBeacon. IBeacon проще с одним типом кадра согласованной длины. Eddystone состоит из четырех различных типов фреймов и имеет переменную длину и форматы кодирования. Обратите внимание, что некоторые поля жестко закодированы, такие как длина, тип и идентификатор компании для iBeacon, а также идентификатор для Eddystone.

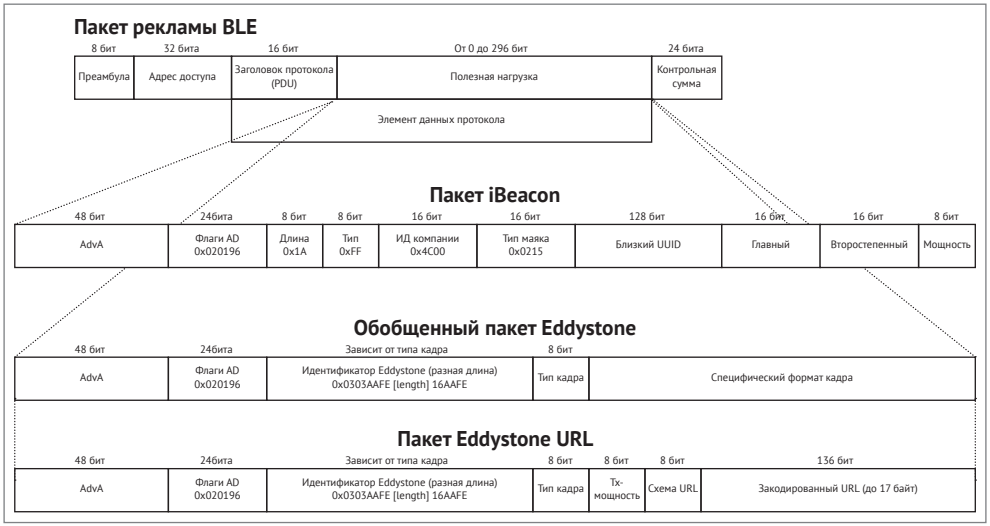


Рис. 5.9 ❖ Пример разницы между рекламными пакетами iBeacon и Eddystone (PDU)

Интервалы сканирования и интервалы рекламы пытаются свести к минимуму количество рекламных объявлений, необходимых для передачи полезных данных в течение определенного периода времени. Окно сканирования, как правило, более длительно, чем реклама, поскольку сканеры, по существу, имеют большую мощность, чем батарейка в маяке. На рис. 5.10 показан процесс рекламы маяка каждые 180 мс, в то время как хост сканирует каждые 400 мс.

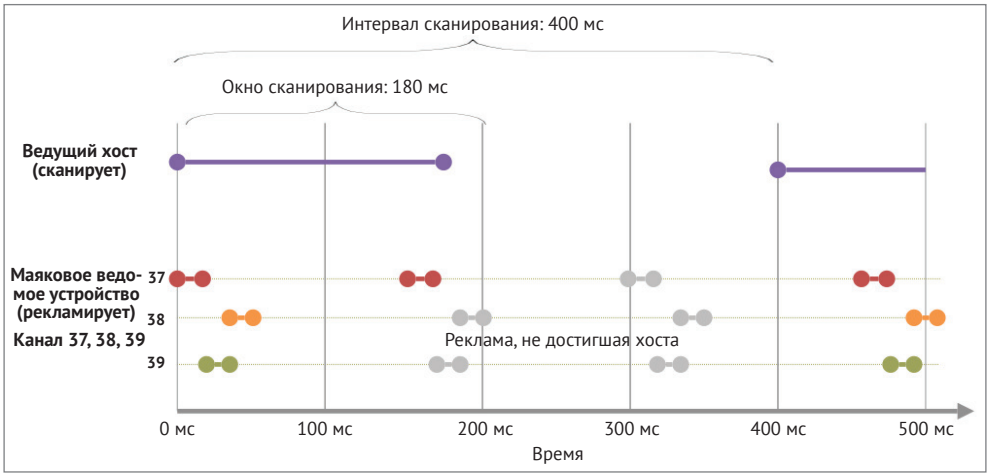


Рис. 5.10 ❖ Пример сканирования хоста с интервалом сканирования 400 мс и окном сканирования 180 мс

Маяк рекламирует каждые 150 мс по выделенным каналам 37, 38, 39. Обратите внимание, что порядок рекламных каналов не является последовательным, так как скачкообразная перестройка частоты может корректировать порядок. Некоторым рекламным объявлениям не удалось связаться с хостом, так как интервал сканирования и интервал рекламы не соответствует фазе. Только одна реклама во втором пакете достигает хоста на канале 37, но по дизайну Bluetooth рекламирует на всех трех каналах, пытаясь максимизировать шансы на успех.

Существует две основные проблемы с архитектурой маяковой системы. Во-первых, это эффект рекламных интервалов в сравнении с точностью отслеживания местоположения. Во-вторых, эффект рекламных интервалов в сравнении с долговечностью батареи, питающей маяк. Оба эффекта уравновешивают друг друга, и требуется тщательная архитектура для правильного развертывания и продления срока службы батареи.

Чем дольше интервал между рекламными сообщениями маяка, тем меньше точность будет у движущейся цели. Например, если розничный торговец отслеживает местоположение посетителей в магазине, а посетители движутся со скоростью ходьбы 4,5 фута в секунду, а набор маяков рекламирует каждые 4 с по сравнению с другим развертыванием, которое рекламирует каждые 100 мс, оно будет показывать разные пути движения к розничному торговцу, собирающему рыночные данные. На рис. 5.11 показаны эффекты медленной и быстрой рекламы в розничном использовании.

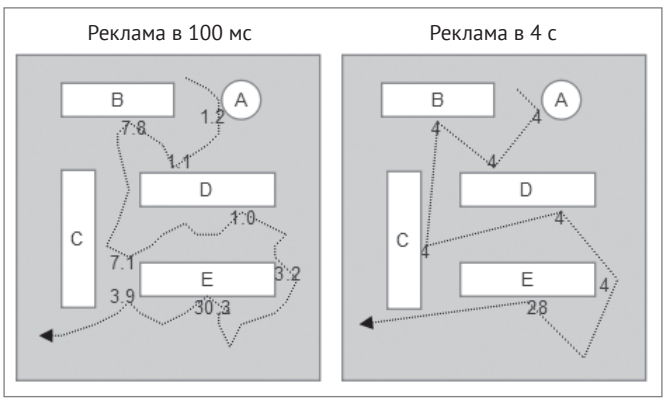


Рис. 5.11 ❖ Эффекты высокочастотной рекламы по сравнению с низкочастотной рекламой на точность определения местоположения. Целые числа представляют собой время клиента, затраченное на определенную точку в магазине

Четырехсекундный рекламный интервал теряет точность при движении клиента по магазину. Кроме того, количество времени, затрачиваемого на определенную точку, можно отслеживать только с дискретными интервалами в 4 с. Время, потраченное на позиции B и C (поскольку клиент покидает мага-

зин), может быть потеряно. В этом случае розничный торговец может захотеть понять, почему клиент потратил 7,8 с в местоположении B и почему клиент вернулся к пункту C на своем пути.

Эффект от частых рекламных сигналов влияет на срок службы батареи маяка. Как правило, батареи в маяках – это литий-ионные ячейки CR2032. *Руководство автостопом по оборудованию iBeacon: всеобъемлющий отчет Aislelabs. Web 14 марта 2016 г.* (https://www.aislelabs.com/reports/beaconguide/) описывает проведение анализа времени автономной работы некоторых обычных маяков и изменение их рекламного интервала (100 мс, 645 мс и 900 мс). Они также использовали различные батареи с увеличением запасенной энергии. Результаты показывают, что средний срок службы колеблется от 0,6 месяцев до более года в зависимости от набора микросхем, но и, что более важно, за счет рекламного интервала. Наряду с рекламным интервалом мощность Tx, безусловно, влияет на общее время автономной работы, но также и количество переданных кадров.

Установка интервалов рекламы слишком длинными, что выгодно для работы от батареи, но плохо для понимания местоположения, имеет вторичный эффект. Если маяк работает в шумной среде и интервал установлен на высокий (> 700 мс), сканер (смартфон) должен будет ждать еще один полный период для приема рекламного пакета. Это может привести к отключению приложений. Быстрый интервал в 100 мс полезен для отслеживания быстродвижущихся объектов (таких как отслеживание активов в логистике флота или сборе информации о маяках с помощью дронов). Если архитектор проектирует систему для отслеживания движения человека с типичной скоростью 4,5 фута в секунду, то от 250 мс до 400 мс является достаточным.

Ценным упражнением для архитектора IoT является понимание стоимости передачи с точки зрения мощности. По сути, устройство IoT имеет ограниченное количество блоков PDU, которые оно может отправлять, прежде чем батарея достигнет точки, где не получится подключить устройство, если не будет восстановлено или пополнено электричество (см. главу 4). Предположим, iBeacon объявляется каждые 500 мс, а длина пакета – 31 байт (это может быть больше). Кроме того, устройство использует аккумуляторную батарею CR2032 с номинальной мощностью 220 мА при 3,7 В. Электроника радиомаяка потребляет 49мкА при 3В. Теперь мы можем предсказать жизнь маяка и эффективность передачи:

- потребление энергии = 49 мкА × 3В = 0,147 мВт;
- байт в секунду = 31 × (1 с/500 мс) × 3 канала = 186 байт/с;
- бит в секунду = 186 байт/с × 8 = 1488 бит/с;
- энергия на бит = 0,147 мВт/(1488 бит/с) = 0,098 мкДж/бит;
- энергия на каждую рекламу = 0,098 мкДж/бит × 31 байт × 8 бит/байт = 24,30 мкДж/рекламу;
- энергия, запасенная в батарее: 220 мАч × 3,7 В × 3,6 с = 2930 Дж;
- жизнь батареи = (2930 Дж× (1 000 000 мкДж/Дж))/((24,30 мкДж/рекламу) × (1 реклама/0,5 с)) × 0,7 = 42 201 646 с = 488 дней = 1,3 года.

Постоянная 0,7 используется для снижения времени автономной работы, как описано в главе 4. 1,3 года являются теоретическим пределом и, скорее всего, не будут получены в полевых условиях из-за таких факторов, как ток утечки, а также другие функции устройства, которые могут потребоваться для периодического запуска.

Заключительная заметка о маяках по отношению к Bluetooth 5. Новая спецификация расширяет длину рекламных сигналов маяка, позволяя передавать рекламные пакеты в каналах передачи данных, а также в каналы рекламы. Это принципиально нарушает предел рекламы, ограничиваясь 31 байтом.

С Bluetooth 5 размер сообщения может быть 255 байт. Новые рекламные каналы Bluetooth 5 называются вторичными рекламными каналами. Они обеспечивают обратную совместимость с унаследованными устройствами Bluetooth 4, определяя конкретный расширенный тип Bluetooth 5 в заголовке. Устаревшие хосты отбрасывают непризнанный заголовок и просто не слушают устройство.

Если хост Bluetooth получает рекламу маяка, указывающую на наличие вторичного рекламного канала, он распознает, что в каналах данных необходимо найти больше данных. Полезная нагрузка первичного рекламного пакета больше не содержит данные маякового радиосигнала, а общую расширенную рекламную полезную нагрузку, которая идентифицирует номер канала данных и смещение по времени. Затем хост будет считывать из этого конкретного канала данные с указанным временным смещением для извлечения фактических данных маякового радиосигнала. Данные могут также указывать на другой пакет (это называется несколькими вторичными рекламными цепями).

Этот новый метод передачи очень длинных сообщений маяка гарантирует, что большие объемы данных могут быть отправлены на клиентские смартфоны. В настоящее время также доступны другие варианты использования и функции, такие как реклама, используемая для передачи синхронных данных, например, аудио-потоки. Маяк может отправлять аудиолекции на смартфон, когда посетитель ходит по музею и смотрит на различные произведения искусства.

Bluetooth 5 также может передавать несколько индивидуальных рекламных объявлений (с уникальными данными и разными интервалами) почти одновременно. Это позволит маякам Bluetooth 5 передавать сигналы Eddystone и iBeacon почти одновременно без какой-либо переконфигурации. Кроме того, маяк Bluetooth 5 может определить, проверен ли он хостом. Это полезно, потому что маяк может определить, получил ли пользователь рекламу, а затем прекратил передачу для экономии энергии.

Диапазон Bluetooth 5 и увеличение скорости

Сила маяка Bluetooth ограничена и может зависеть от ограничений, установленных на мощность передатчика, чтобы сохранить срок службы батареи. Обычно для оптимального диапазона маяков и мощности сигнала необходима линия прямой видимости. На рис. 5.12 приведена сила сигнала для кривой расстояния для типичной связи Bluetooth 4.0 4.0.

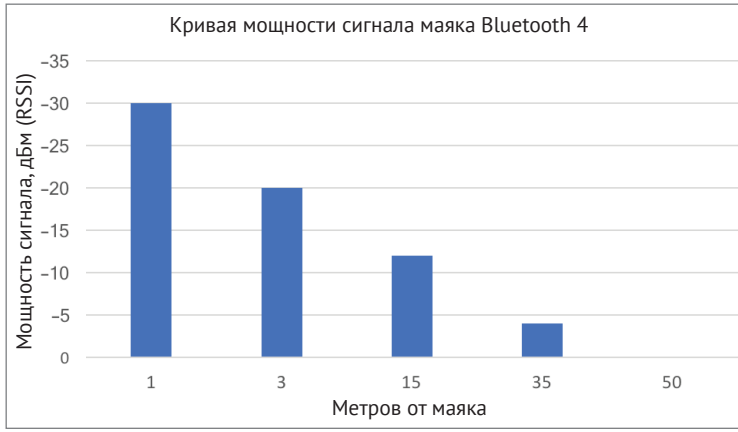


Рис. 5.12 ❖ Сила маяка ограничена.
Часто производитель ограничивает мощность Tx маяка, чтобы сохранить время работы от батареи. При удалении от маяка уровень сигнала падает, как ожидалось. Обычно 30-футовый диапазон – это полезное расстояние маяка (Bluetooth 4.0)

Bluetooth также имеет разные уровни мощности, диапазоны и мощность передачи, основанные на классификации для каждого устройства, как показано в табл. 5.2.

Таблица 5.2. Классы Bluetooth

Номер класса	Максимальный выходной уровень (дБм)	Максимальная выходная мощность (мВт)	Максимальная дальность	Использование
1	20	100	100 м	Адаптер USB, точки доступа
1, 2	10	10	30 м (обычно 5 м)	Маяки, носимая электроника
2	4	2,5	10 м	Мобильные устройства, адаптеры Bluetooth, считыватели смарт-карт
3	0	1	10 см	Адаптеры Bluetooth

Bluetooth 5 улучшил диапазон, а также скорость передачи данных за пределы устаревших ограничений Bluetooth. Новая радиостанция PHY доступна с Bluetooth 5 под названием LE2M. Это удваивает скорость передачи необработанных данных Bluetooth с 1 Мсимвол/с до 2 Мсимвол/с. Для передачи равного количества данных по Bluetooth 5, в отличие от Bluetooth 4, для передачи потребуется меньше времени. Это особенно актуально для устройств IoT, работающих на батарейках. Новый PHY также увеличивает мощность от +10 дБм до +20 дБм, что позволяет увеличить дальность.

Что касается диапазона, у Bluetooth 5 есть еще одна опция PHY для передачи расширенного диапазона в BLE. Этот вспомогательный PHY имеет *кодировку*

LE. Этот PHY по-прежнему использует скорость 1 Мсимвол/с, как в Bluetooth 4.0, но снижает более низкую кодировку пакетов 125 Кбит/с или Кбит/с и увеличивает мощность передачи на +20 дБм. Это приводит к увеличению диапазона в 4 раза по сравнению с Bluetooth 4.0 и лучшему проникновению в здания. LE-кодированный PHY увеличивает потребление энергии в интересах увеличения диапазона.

Введение в mesh-сеть Bluetooth

После того, как была выпущена спецификация Bluetooth 5, SIG сосредоточился на формализации mesh-сети Bluetooth. Bluetooth SIG опубликовал профиль mesh-сети, устройство и спецификацию модели 1.0 13 июля 2017 г. Это происходит через шесть месяцев после выпуска спецификации Bluetooth 5.0. До официальной спецификации Bluetooth 5 у Bluetooth SIG существовали собственные и специальные схемы, использующие более старые версии Bluetooth для создания mesh-сред. Три спецификации, опубликованные Bluetooth SIG, следующие:

- **спецификация профиля Mesh 1.0** – определяет основные требования, позволяющие интегрированное решение mesh-сети;
- **спецификация модели Mesh 1.0** – базовая функциональность узлов в mesh-сети;
- **свойства устройства Mesh 1.0** – определяет свойства устройства, необходимые для спецификации модели mesh-сети.

Пока неизвестно, существует ли какой-либо предел размера mesh-сети. В спецификации есть некоторые ограничения. По спецификации 1.0 может быть до 32 767 узлов в mesh-сети Bluetooth и 16384 физических группах. Максимальное время жизни, которое указывает на глубину mesh-сети, составляет 127.

Mesh-сеть Bluetooth 5 теоретически допускает наличие 2^{128} виртуальных групп. Практически группировка будет гораздо более ограничена.

Mesh-сеть Bluetooth основана на BLE и располагается на физическом и канальном уровнях BLE, описанном ранее. Сверху этого уровня находится стек уровней, специфичных для mesh-сети:

- **модели** – реализует поведение, состояния и привязки по одной или нескольким спецификациям модели;
- **основные модели** – настройка и управление mesh-сетью;
- **уровень доступа** – определяет формат данных приложения, процесс шифрования и проверку данных;
- **верхний транспортный уровень** – управляет аутентификацией, шифрованием и расшифровкой данных, передаваемых на уровень доступа и с него. Транспортирует управляющие сообщения, такие как дружба и сердцебиение;
- **нижний транспортный уровень** – выполняет **сегментацию и повторную сборку (SAR)** фрагментированных блоков PDU, если это необходимо;

- **сетевой уровень** – определяет, какой сетевой интерфейс выводит сообщения. Управляет различными типами адресов и поддерживает множество носителей;
- **уровень носителя** – определяет, как обрабатываются PDU. Поддерживаются два типа PDU: рекламный носитель и носитель GATT. Рекламный носитель обрабатывает передачу и прием PDU mesh-сети, в то время как носитель GATT предоставляет прокси-сервер для устройств, которые не поддерживают рекламный носитель;
- **BLE** – полная спецификация Bluetooth LE.

Mesh-сеть Bluetooth может включать другие mesh-сети или функции BLE. Устройство, способное поддерживать mesh-сети и BLE, может связываться с другими устройствами, такими как смартфоны, или иметь функции маяка. На рис. 5.13 показан стек mesh-сети Bluetooth. Важно понимать замену стека выше канального уровня.

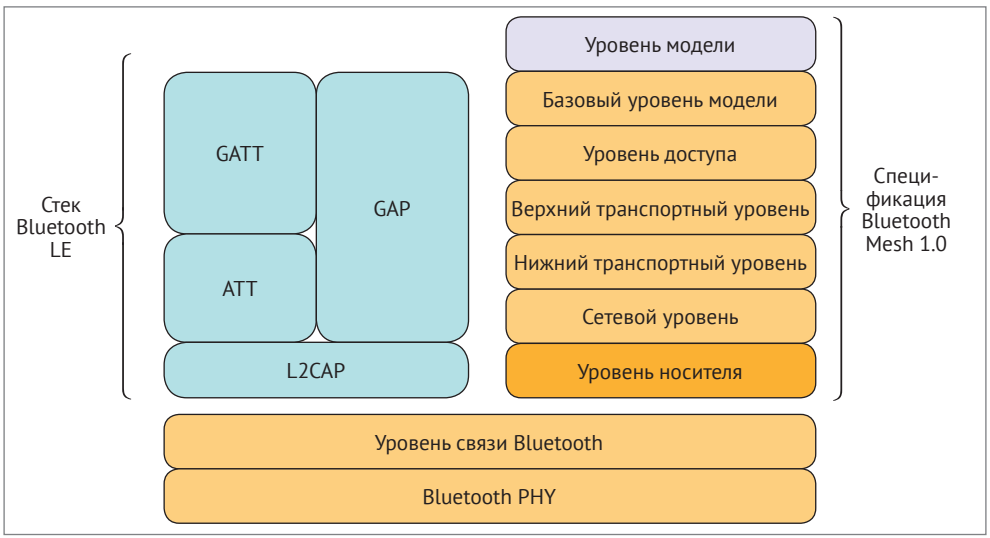


Рис. 5.13 ❖ Спецификация 1.0 стека mesh-сети Bluetooth

Топология mesh-сети Bluetooth

Mesh-сеть Bluetooth использует концепцию сети наводнений. В сети наводнения каждый входящий пакет, полученный узлом в сетке, отправляется через каждую исходящую связь, за исключением ссылки на родителя сообщения. Преимущество наводнений заключается в том, что, если пакет может быть доставлен, он будет доставлен (хотя, вероятно, много раз через многие маршруты). Он автоматически найдет самый короткий маршрут (который может изменяться по качеству сигнала и расстоянию в динамической mesh-сети). Алгоритм является самым простым в реализации протоколов маршрутизации. Кроме того,

он не требует центрального менеджера, такого как сеть Wi-Fi, основанная на центральном маршрутизаторе. Для сравнения, альтернативные типы маршрутизации mesh-сети включают в себя алгоритмы на основе дерева. С помощью алгоритмов дерева (или алгоритмов кластерного дерева) координатор необходим для создания экземпляра сети и становится родительским узлом. Однако деревья не обязательно являются истинными mesh-сетями. Другие протоколы маршрутизации mesh-сетей включают проактивную маршрутизацию, которая поддерживает современные таблицы маршрутизации на каждом узле и реактивную маршрутизацию, которая только обновляет таблицы маршрутизации на каждом узле по требованию; например, когда данные должны отправляться через узел. Zigbee (рассматриваемый позже) представляет собой форму проактивной маршрутизации, называемую **Ad Hoc On-Demand Distance Vector (AODV)**. Рисунок 5.14 иллюстрирует потоковое вещание. Время прибытия на каждом уровне может динамически меняться от узла к узлу. Кроме того, mesh-сеть должна быть устойчивой для дублирования сообщений, поступающих на любой узел, как в случае узла *7* и узла *П*.

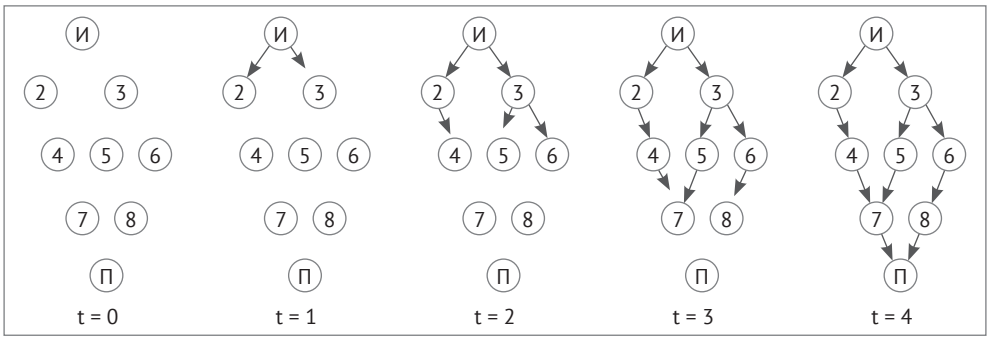


Рис. 5.14 ❖ Архитектура наводнения в mesh-сети.
И = Источник, П = Получатель. Источник производит данные,
которые распространяются и протекают через каждый узел в сетке

Основным недостатком сети наводнений является пропускная способность. В зависимости от вентилятора от каждого узла, перегрузка в mesh-сети Bluetooth может быть значительной. Другая проблема – атака отказа в обслуживании. Если mesh-сеть просто отключает сообщения, необходимо знать, когда прекращать передачу. Bluetooth выполняет это через идентификаторы времени жизни, которые мы рассмотрим позже.

Узлы в mesh-сети Bluetooth включают следующее:

- **узлы** – это устройства Bluetooth, которые были предварительно подготовлены и являются членами mesh-сети;
- **неподготовленные устройства** – это устройства, которые могут присоединиться к mesh-сети, которые еще не является ее частью и не были подготовлены;

- **элементы** – узел с несколькими составными частями. Каждая часть может контролироваться и адресоваться независимо. Примером может служить узел Bluetooth с датчиками температуры, влажности и люмен. Это будет единственный узел (датчик) с тремя элементами;
- **шлюз mesh-сети** – узел, который может переводить сообщения между mesh-сетью и технологией не-Bluetooth.

После подготовки узел может поддерживать необязательный набор функций, которые включают:

- **реле** – узел, поддерживающий реле, называется узлом ретрансляции и может повторно передавать полученные сообщения;
- **прокси** – позволяет устройствам Bluetooth LE, которые не поддерживают mesh-сети Bluetooth изначально, взаимодействовать с узлами в mesh-сети. Это выполняется с использованием прокси-узла. Прокси-сервер предоставляет интерфейс GATT с устаревшим устройством Bluetooth, и определяется прокси-протокол, основанный на канале, ориентированном на соединение. Унаследованное устройство считывает и записывает протокол прокси-сервера GATT, а прокси-узел преобразует сообщения в настоящие PDU mesh-сети;
- **малая мощность** – некоторые узлы в mesh-сети должны иметь чрезвычайно низкие уровни энергопотребления. Они могут обслуживать информацию об экологических датчиках (например, температуру) один раз в час и настраиваться с помощью управляющего узла или облака один раз в год. Этот тип устройства не может быть помещен в режим прослушивания, когда сообщение будет поступать только один раз в год. Узел входит в роль, называемую **узлом низкой мощности (LPN)**, который соединяет его с дружественным узлом. LPN переходит в состояние глубокого сна и опроса связанного друга для любых сообщений, которые могли появиться во время сна;
- **друг** – дружественный узел связан с LPN, но не обязательно ограничивается мощностью, как LPN. Друг может использовать выделенную цепь или силу стены. Обязанность друга состоит в том, чтобы хранить и буферизовать сообщения, предназначенные для LPN, до тех пор, пока LPN не просыпается и не опросит его для сообщений. Многие сообщения могут быть сохранены, и друг передаст их в порядке, используя флаг **дополнительных данных (MD)**.

Рисунок 5.15 иллюстрирует топологию сети Bluetooth с различными компонентами, поскольку они будут связаны друг с другом в реальной mesh-сети.

Mesh-сеть Bluetooth будет кэшировать сообщения на каждом узле; это имеет решающее значение в сети наводнений. Поскольку одно и то же сообщение может поступать в разное время из разных источников, кэш обеспечивает поиск последних сообщений, полученных и обработанных. Если новое сообщение идентично одному в кэше, оно отбрасывается. Это обеспечивает идемпотентность системы.

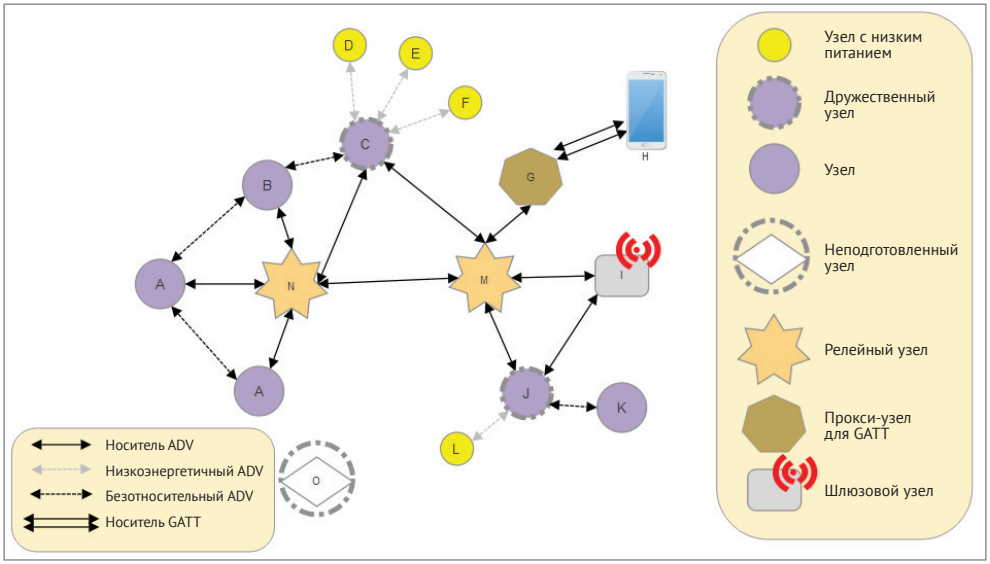


Рис. 5.15 ❖ Топология mesh-сети Bluetooth. Обратите внимание на классы, включая узлы, LPN, друзей, шлюзы, релейные узлы и неподготовленные узлы

Каждое сообщение содержит поле **времени для жизни (TTL)**. Если сообщение получено узлом и затем повторно передается, TTL уменьшается на единицу. Это механизм безопасности для предотвращения бесконечных циклов, и сети создают усиление атак типа «отказ в обслуживании» внутри mesh-сети.

Сообщение о сердцебиении периодически передается от каждого узла в mesh-сети. Сердцебиение сообщает среде, что узел все еще присутствует и здоров. Он также позволяет mesh-сети знать, как далеко находится узел и изменилось ли расстояние с момента последнего биения. По сути, он подсчитывает количество переходов для достижения узла. Этот процесс позволяет mesh-сети реорганизовываться и самовосстанавливаться.

Режимы адресации mesh-сети Bluetooth

Mesh-сеть Bluetooth использует три формы адресации:

- **одноадресная адресация** – уникально идентифицирует один элемент в mesh-сети. Адрес назначается в процессе подготовки;
- **групповая адресация** – это форма многоадресной адресации, которая может представлять один или несколько элементов. Они либо предопределены идентификатором Bluetooth SID как фиксированные групповые адреса SIG, либо назначены «на лету»;
- **виртуальная адресация** – адрес может быть назначен более чем одному узлу и нескольким элементам. Виртуальная адресация использует 128-битный UUID. Предполагается, что UUID может быть настроен про-

изводителем, чтобы позволить ему обращаться к своему продукту во всем мире.

Протокол mesh-сети Bluetooth начинается с 384-байтовых сообщений, которые сегментируются в 11-байтовые посылки. Вся связь в mesh-сети Bluetooth ориентирована на сообщения. Существуют две формы сообщений, которые могут быть переданы:

- **подтвержденные сообщения** – для них требуется ответ от узла (узлов), которые получают сообщение. Подтверждение также включает данные, которые отправитель запрашивает в исходном сообщении. Поэтому это подтвержденное сообщение служит двум целям;
- **неподтвержденные сообщения** – они не требуют ответа от получателя.

Отправка сообщения с узла также известна как публикация. Узлы, настроенные для обработки избранных сообщений, отправленных на определенные адреса, называются подписчиками. Каждое сообщение зашифровывается и аутентифицируется с использованием сетевого ключа и ключа приложения.

Ключ приложения относится к приложению или случаю использования (например, включение света в зависимости от настройки цвета светодиода). Узлы будут публиковать события (световые переключатели), другие узлы будут подписываться на эти события (лампы и лампочки). На рис. 5.16 показана топология сети Bluetooth. Здесь узлы могут подписаться на несколько событий (освещение в лобби и освещение прихожей). Круги представляют собой групповые адреса. Коммутатор опубликует для группы.

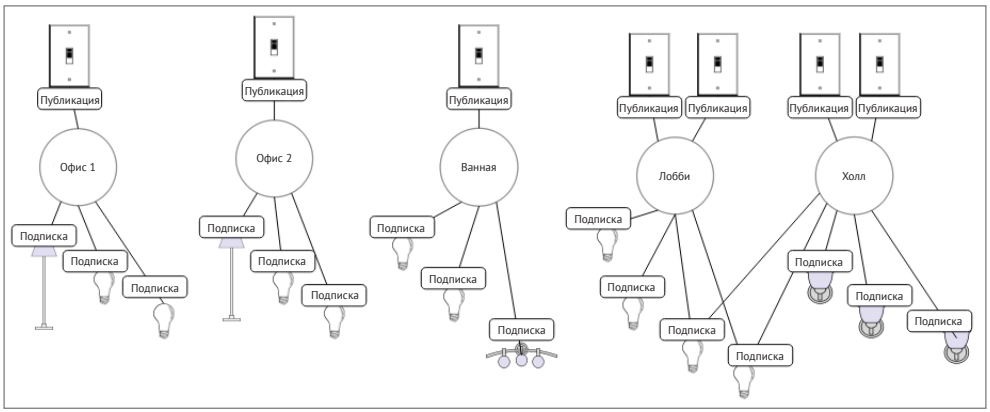


Рис. 5.16 ❖ Модель публикация – подписчик mesh-сети Bluetooth

Mesh-сеть Bluetooth представляет концепцию группового обмена сообщениями. В mesh-сети у вас может быть группировка похожих объектов, таких как освещение в ванной или освещение в лобби. Это помогает в удобстве использования; например, если добавлен новый свет, только этот свет

должен быть подготовлен, а остальная часть mesh-сети не нуждается в каких-либо изменениях.

Переключатели света в предыдущем примере имеют два состояния: включенный и выключенный. Mesh-сеть Bluetooth определяет состояния, и в этом случае они помечены общим OnOff. Общие состояния и сообщения поддерживают многие типы устройств от источников света до вентиляторов на приводы. Это быстрый способ повторного использования модели для общей цели. Когда система переходит из одного состояния в другое, это называется переходом состояния в mesh-сети. Состояния также могут быть связаны друг с другом. Если состояние изменяется, оно может повлиять на переход в другое состояние. В качестве примера, mesh-сеть, управляющая потолочным вентилятором, может иметь состояние управления скоростью, которое, когда оно достигает значения нуля, изменяет общее состояние включения-выключения.

Свойства аналогичны состояниям, но имеют больше, чем двоичные значения. Например, температурный датчик может иметь состояние температуры 8 для обозначения и публикации восьмибитового значения температуры. Свойство может быть установлено как производителем (только для чтения) – поставщиком элемента или администратором, который имеет доступ на чтение и запись. Оба состояния и свойства передаются через сообщения в mesh-сети. Сообщения бывают трех типов:

- **запросить** – запрашивает значение заданного состояния из одного или нескольких узлов;
- **установить** – изменяет значение состояния;
- **статус** – есть ответ на запрос, который содержит данные.

Все эти концепции состояний и свойств в конечном итоге образуют модель (самый высокий уровень стека mesh-сети Bluetooth). Модель может быть сервером, и в этом случае он определяет состояния и переход состояний. Альтернативно, модель может быть клиентом, который не определяет какие-либо состояния; скорее, он определяет сообщения взаимодействия с состоянием для использования с запросом, установкой и статусом. Модель управления может поддерживать гибрид моделей серверов и клиентов.

Mesh-сеть также может использовать стандартные функции Bluetooth 5, такие как анонимные рекламные объявления и множество рекламных наборов. Возьмите случай с mesh-сетью, которая соединяется с телефоном для голосовой связи, но одновременно передает пакеты для других целей. Используя несколько наборов объявлений, оба варианта использования могут управляться одновременно.

Подготовка mesh-сети Bluetooth

Узел может присоединяться к mesh-сети посредством действия инициализации. Подготовка – это безопасный процесс, который принимает неподготовленное и небезопасное устройство и преобразует его в узел в mesh-сети. Сначала узел будет защищать NetKey от mesh-сети. По крайней мере, один NetKey должен быть на каждом устройстве в mesh-сети для соединения. Устройства

добавляются в mesh-сеть через средство подготовки. Подготовитель распределяет сетевой ключ и уникальный адрес на неподготовленное устройство. В процессе инициализации используется обмен ключами Диффи-Хеллмана с эллиптической кривой для создания временного ключа для шифрования сетевого ключа. Это обеспечивает безопасность от атаки «человек-в-середине» во время подготовки. Ключ устройства, полученный из эллиптической кривой, используется для шифрования сообщений, отправленных из подготовителя на устройство.

Процесс подготовки следующий:

1) неподготовленное устройство транслирует рекламный пакет маяка;
2) подготовитель отправляет приглашение на устройство. Неподготовленное устройство отвечает возможностями PDU для обеспечения доступа;
3) подготовитель и устройства обмениваются открытыми ключами;
4) неподготовленное устройство выводит произвольное число пользователю. Пользователь вводит цифры (или идентификатор) в подготовитель, и криптографический обмен начинает завершать фазу аутентификации;
5) ключ сеанса выводится каждым из двух устройств из закрытого ключа и обменными открытыми ключами. Ключ сеанса используется для защиты данных, необходимых для завершения процесса подготовки, включая обеспечение безопасности NetKey;
6) устройство изменяет состояние с неподготовленного устройства на узел и теперь обладает NetKey, одноадресным адресом и параметром безопасности mesh-сети, называемым индексом IV.

IEEE 802.15.4

IEEE 802.15.4 – это стандартная беспроводная личная сеть, определенная рабочей группой IEEE 802.15. Модель была ратифицирована в 2003 г. и составляет основу многих других протоколов, включая Thread (описываемый позже), Zigbee (описываемый позже), Wireless HART и другие. 802.15.4 определяет только нижнюю часть (PHY и канальный уровень) стека, а не верхние уровни. Создание полного сетевого решения зависит от других консорциумов и рабочих групп. Целью 802.15.4 и протоколов, которые находятся на нем, является недорогой WPAN с низким энергопотреблением. Последняя спецификация – это спецификация IEEE 802.15.4e, ратифицированная 6 февраля 2012 г., которая является версией, которую мы обсудим в этой главе.

Архитектура IEEE 802.15.4

Протокол IEEE 802.15.4 работает в нелицензированном спектре в трех разных диапазонах радиочастот: 868 МГц, 915 МГц и 2400 МГц. Цель состоит в том, чтобы иметь как можно более широкое географическое распространение, что подразумевает три разных диапазона и несколько методов модуляции. Хотя более низкие частоты позволяют 802.15 иметь меньше проблем с радиочастотными интерференциями или диапазоном, полоса 2,4 ГГц является, безусловно, са-

мой часто используемой 802.15.4 во всем мире. Более высокая полоса частот приобрела свою популярность, поскольку более высокая скорость позволяет использовать более короткие рабочие циклы при передаче и приеме, тем самым сохраняя энергию.

Другим фактором, который сделал популярной полосу 2,4 ГГц, является ее признание на рынке благодаря популярности Bluetooth. В табл. 5.3 приведены различные методы модуляции, географическая область и скорость передачи данных для различных диапазонов 802.15.4.

Таблица 5.3. Методы модуляции, географическая область и скорость передачи данных для различных диапазонов 802.15.4

Диапазон частот (МГц)	Номера каналов	Модуляция	Скорость передачи данных (Кбит/с)	Регион
868,3	1 канал: 0	BPSK O-QPSK ASK	20 100 250	Европа
902–928	10 каналов: 1–10	BPSK O-QPSK ASK	40 250 250	Северная Америка, Австралия
2405–2480	16 каналов: 11–26	O-QPSK	250	Весь мир

Типичный диапазон протокола 802.15.4 составляет около 200 метров в открытом режиме, с прямой видимостью. В помещении типичный диапазон составляет около 30 м. Для расширения диапазона можно использовать трансиверы большей мощности (15 дБм) или mesh-сети. На рис. 5.17 показаны три полосы, используемые 802.15.4 и распределение частот.

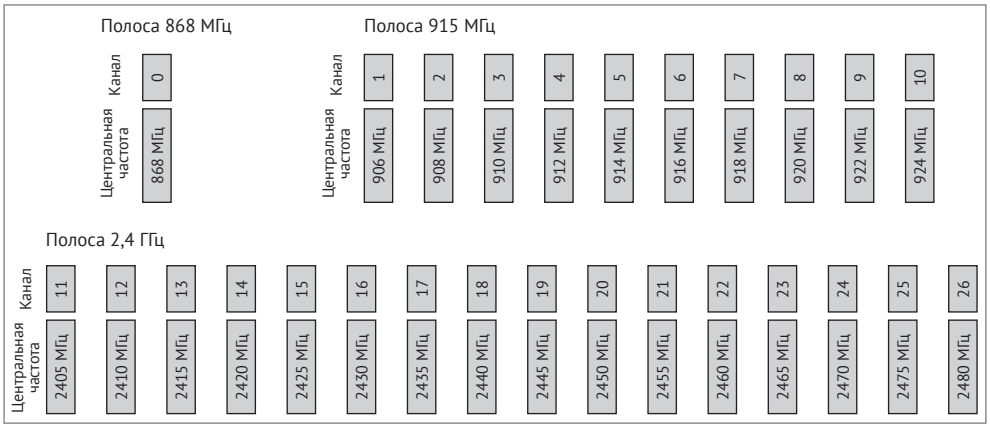


Рис. 5.17 ❖ IEEE 802.15.4 и распределения частот.
915 МГц используется разделение частот 2 МГц,
а в полосе 2,4 ГГц используется разделение частот 5 МГц

Для управления общим частотным пространством 802.15.4 и большинство других беспроводных протоколов используют некоторую форму **предотвращения столкновений с множественным доступом с поддержкой несущей (CSMA/CA)**. Поскольку невозможно прослушивать канал при передаче по тому же каналу, схемы обнаружения столкновений не работают; поэтому мы используем предотвращение столкновений. CSMA/CA просто прослушивает определенный канал в течение заданного промежутка времени. Если канал воспринимается «холостым», то он передает сначала сигнал, сообщающий всем другим передатчикам, что канал занят. Если канал занят, передача будет отложена на случайный период времени. В закрытой среде CSMA/CA обеспечит 36%-ное использование канала; однако в реальных сценариях только 18% каналов будут использоваться.

Группа IEEE 802.15.4 определяет, что рабочая мощность передачи должна быть не менее 3 дБм, а чувствительность приемника –85 дБм на частоте 2,4 ГГц и –91 дБм на частоте 868/915 МГц. Как правило, это подразумевает ток от 15 мА до 30 мА для передачи и ток от 18 мА до 37 мА для приема.

Скорость передачи данных в виде состояний достигает 250 кбит/с, как указано, с помощью квадратурного сдвигового фазового смещения.

Стек протокола состоит только из двух нижних уровней модели OSI (PHY и MAC). PHY отвечает за кодирование символов, битовую модуляцию, демодуляцию бит и синхронизацию пакетов. Он также выполняет переключение режима приема-передачи и управление задержкой времени/подтверждения задержки внутри пакета. На рис. 5.18 приведен стек протоколов 802.15.4 как сравнение с моделью OSI.

Рис. 5.18 ❖ Стек протокола IEEE 802.15.4. Определены только уровни PHY и MAC, другие стандарты и организации могут включать уровни с 3 по 7 выше PHY и MAC

Сверху физического уровня находится канальный уровень, ответственный за обнаружение и исправление ошибок на физическом линке. Этот уровень

также управляет уровнем доступа к среде (MAC) для обработки предотвращения столкновений с использованием таких протоколов, как CSMA/CA. Уровень MAC обычно реализуется в программном обеспечении и работает на MCU, таком как популярный ARM Cortex-M3 или даже 8-битный ATmega. Есть несколько поставщиков микросхем, которые включили MAC в чистый набор микросхем, например, Microchip Technology Inc.

Интерфейс от MAC до верхних уровней стека обеспечивается через два интерфейса, называемые **сервисными точками доступа (SAP)**:

- **MAC-SAP** – для управления данными;
- **MLME-SAP** – для управления и мониторинга (объект управления уровнем MAC).

В IEEE 802.15.4 есть два типа связи: маяк и безмаяковая связь.

Для сети с маяком уровень MAC может генерировать маяки, которые позволяют устройству вводить PAN, а также предоставлять события синхронизации для устройства, чтобы войти в канал связи. Маяк также используется для аккумуляторных устройств, которые обычно спят. Устройство просыпается на периодически и слушает маяк от своих соседей. Если маяк слышен, он начинает фазу, называемую интервалом SuperFrame, где временные интервалы предварительно распределены, чтобы гарантировать пропускную способность для устройств, а устройства могут вызвать внимание соседа. **Интервал SuperFrame (SO)** и **интервал маяка (BO)** полностью контролируются координатором PAN. SuperFrame разделен на шестнадцать одинаковых временных интервалов с одним выделенным в качестве маяка этого SuperFrame. Канальный доступ к каналу CSMA/CA используется в сетях с радиомаяками.

Гарантированные временные интервалы (GTS) могут быть назначены определенным устройствам, предотвращающим любые разногласия. Допускается до семи доменов GTS. Слоты GTS выделяются координатором PAN и объявляются в маяке, который он передает. Координатор PAN может изменять распределение GTS «на лету» динамически, исходя из нагрузки, требований и емкости системы. Направление GTS (передача или прием) предопределено до начала GTS. Устройство может запросить один передающий и/или один принимающий GTS.

В SuperFrame есть **периоды доступа к конкуренции (CAP)**, где есть перекрестные помехи на канале и **свободные периоды (CFP)**, где кадр может использоваться для передачи и GTS. На рис. 5.19 показан суперкадр, состоящий из 16 равных временных интервалов, ограниченных сигналами маяка (один из которых должен быть маяком). Свободный от конфликта период будет далее разделен на гарантированные временные интервалы (GTS), и один или несколько GTSW могут быть выделены конкретному устройству. Никакое другое устройство не может использовать этот канал во время GTS.

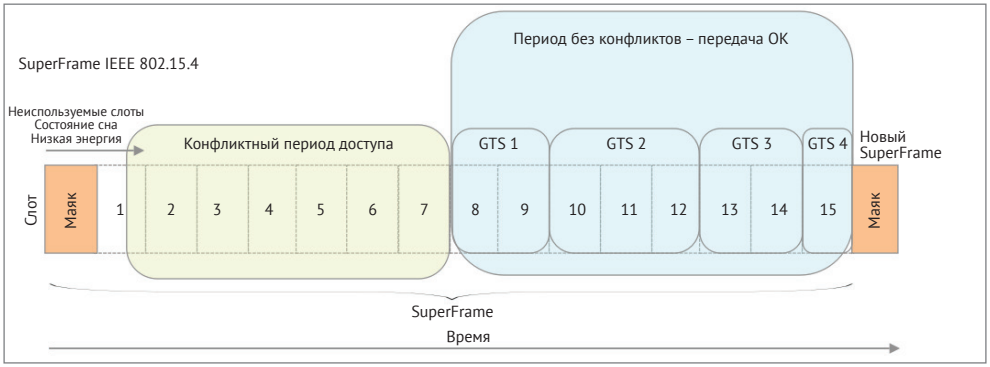


Рис. 5.19 ❖ Последовательность IEEE 802.15.4 SuperFrame

В дополнение к сети на основе радиомаяков IEEE 802.15.4 позволяет создавать сети без радиомаяка. Это гораздо более простая схема, при которой кадр маякового радиосигнала не передается координатором PAN. Это означает, однако, что все узлы находятся в режиме приема все время. Это обеспечивает постоянный конкурентный доступ с использованием нераспределенного CSMA/CA. Передающий узел будет выполнять четкую оценку канала (CCA), в которой он прослушивает канал, чтобы определить, используется ли он, а затем передается, если он очищен. CCA является частью алгоритма CSMA/CA и используется для «ощущения», если используется канал. Устройство может получать доступ к каналу, если оно не отличается трафиком от других устройств (включая устройства, не поддерживающие 802.15.4). В случае, когда канал занят, алгоритм переходит на «отключаемый» алгоритм и ждет случайный промежуток времени, чтобы повторить CCA. Группа IEEE 802.15.4 определяет для использования CCA следующее:

- **режим CCA 1** – энергия выше порога (нижайшего). CCA сообщит занятой среде об обнаружении любой энергии, превышающей пороговое значение (ED);
- **режим CCA 2** – только для обнаружения несущей (средний – по умолчанию). Этот режим заставляет CCA сообщать о занятом носителе только в том случае, если обнаружен сигнал с **расширенным спектром прямой последовательности (DSSS)**. Сигнал может быть выше или ниже порога ED;
- **режим CCA 3** – обнаружение несущей с энергией выше порога (наибольшего). В этом режиме CCA сообщается о занятости, если он обнаруживает сигнал DSSS с энергией выше порога ED;
- **режим CCA 4** – этот режим является режимом обнаружения несущей с таймером. CCA запускает таймер на некоторое количество миллисекунд и будет сообщать о занятости только в том случае, если он обнаруживает высокоскоростной PHY-сигнал. CCA сообщит, что среда простаивает, если таймер истекает, и не наблюдается высокоскоростной сигнал;

- **режим CCA 5** – это комбинация обнаружения несущей и энергии выше порогового режима.

Насчет режимов CCA:

- обнаружение энергии будет не более чем на 10 дБ выше указанной чувствительности приемника;
- время обнаружения CCA будет равно восьми периодам символа;
- следует отметить, что режимы CCA, основанные на обнаружении энергии, потребляют наименьшее количество энергии для устройства.

Этот режим будет потреблять гораздо больше энергии, чем связь с радиомаяком.

Топология IEEE 802.15.4

В IEEE 802.15.4 есть два основных типа устройств:

- **полнофункциональное устройство (FFD)** – поддерживает любую топологию сети, может быть координатором сети (PAN) и может связываться с любым устройством PAN-координатором;
- **ограниченное по функциональности устройство (RFD)** – ограничено только топологией звезды, не может выполнять функции координатора сети, может связываться только с сетевым координатором.

Звездная топология является самой простой, но требует, чтобы все сообщения между одноранговыми узлами проходили через координатор PAN для маршрутизации. Одноранговая топология является типичной mesh-сетью и может напрямую связываться с соседними узлами. Для построения более сложных сетей и топологий требуется протокол более высокого уровня, который мы обсудим в разделе *Zigbee*.

Координатор PAN имеет уникальную роль, которая заключается в настройке и управлении PAN. Он также обязан передавать сетевые маяки и сохранять информацию об узле. В отличие от датчиков, которые могут использовать аккумуляторы или источники энергии, координатор PAN постоянно принимает передачи и обычно находится на выделенной линии электропитания (от розетки). Координатор PAN всегда является FFD.

RFD или даже маломощные FFD могут работать на аккумуляторах. Их роль заключается в поиске доступных сетей и передаче данных по мере необходимости. Эти устройства могут быть переведены в состояние ожидания в течение очень длительных периодов времени. На рис. 5.20 приведена диаграмма звездной топологии в сравнении с одноранговой.

В пределах PAN разрешены широковещательные сообщения. Чтобы транслировать всю среду, нужно указать только идентификатор PAN 0xFFFF.

Режимы адресации IEEE 802.15.4 и структура пакета

Стандарт определяет, что все адреса основаны на уникальных 64-битных значениях (адрес IEEE или MAC-адрес). Однако для экономии полосы пропускания и уменьшения энергии передачи таких больших адресов 802.15.4 позволяет устройству, соединяющему сеть, «обменивать» свой уникальный 64-разряд-

ный адрес на короткий 16-битный локальный адрес, что позволяет повысить эффективность передачи и дает большее энергосбережение.

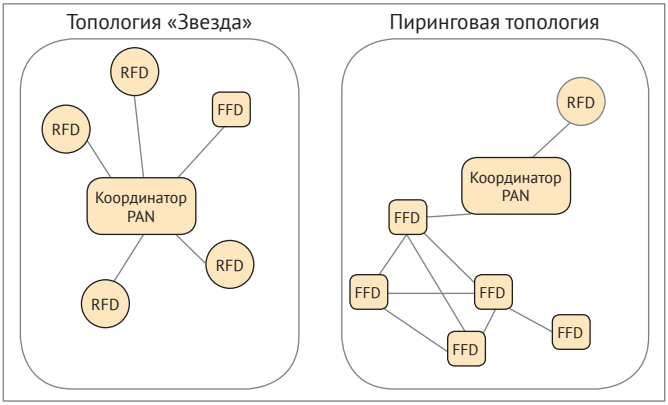


Рис. 5.20 ❖ Руководство IEEE 802.15.4 по сетевым топологиям. Внедряющий 802.15.4 может свободно создавать другие сетевые топологии

Ответственность за этот «обменный» процесс несет координатор PAN. Мы называем этот 16-битный локальный адрес идентификатором PAN. Сама сеть PAN имеет идентификатор PAN, поскольку могут существовать несколько PAN. На рис. 5.21 показана схема структуры пакетов 802.15.4.

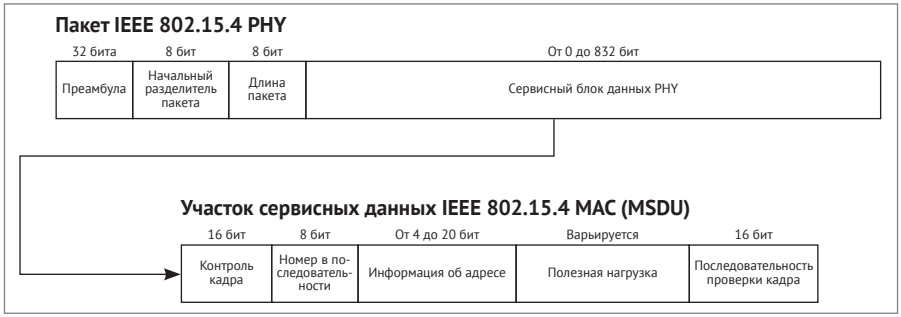


Рис. 5.21 ❖ IEEE 802.15.4: кодирование пакетов PHY и MAC

Кадры являются базовой единицей переноса данных, и существует четыре основных типа (некоторые из основных понятий были рассмотрены в последнем разделе):

- **кадр данных** – транспорт данных приложений;
- **кадр подтверждения** – подтверждение приема;
- **кадр маяка** – отправлено координатором PAN для создания структуры SuperFrame;

- **командный кадр MAC** – управление уровнем MAC (ассоциировать, дизассемблировать, запрос маяка, запрос GTS).

Последовательность запуска IEEE 802.15.4

IEEE 802.15.4 поддерживает процесс запуска, конфигурации сети и объединения существующих сетей. Процесс выглядит следующим образом:

1) устройство инициализирует свой стек (уровни PHY и MAC);
2) создан координатор PAN. Каждая сеть имеет только один координатор PAN. Координатор PAN должен быть назначен на этом этапе, прежде чем продолжать;
3) координатор PAN будет слушать другие сети, к которым он имеет доступ, и получает идентификатор PAN, который является уникальным для PAN, которым он будет управлять. Он может делать это по нескольким частотным каналам;
4) координатор PAN выберет конкретную радиочастоту для использования в сети. Он будет делать это с помощью обнаружения энергии, сканируя частоты, которые может поддерживать PHY и слушает, чтобы найти тихий канал;
5) сеть будет запущена путем настройки координатора PAN и последующего запуска устройства в режиме координатора. На этом этапе координатор PAN может принимать запросы;
6) узлы могут подключаться к сети путем поиска координатора PAN с использованием сканирования активного канала, где он передает запрос маяка по всем его частотным каналам. Когда координатор PAN обнаружит маяк, он ответит обратно запрашивающему устройству. В качестве альтернативы, в сети с маяком (подробно описанной ранее) координатор PAN будет обычно отправлять маяковый радиосигнал, и устройство может выполнять сканирование пассивного канала и прослушивать маяковый радиосигнал. Затем устройство отправит запрос об ассоциации;
7) координатор PAN определит, должно ли устройство подключаться к сети. Это может быть основано на правилах управления доступом или даже на том есть ли у координатора PAN достаточно ресурсов для управления другим устройством. Если это принято, координатор PAN назначит 16-разрядный короткий адрес устройству.

Безопасность IEEE 802.15.4

Стандарт IEEE 802.15.4 включает положения о безопасности в виде шифрования и аутентификации. Архитектор обладает гибкостью в обеспечении безопасности сети на основе затрат, производительности, безопасности и мощности. Различные варианты безопасности перечислены в табл. 5.4.

Шифрование на основе AES использует блочный шифр с режимом счетчика. AESCBC-MAC обеспечивает защиту только для проверки подлинности, а режим AES-CCM обеспечивает полный набор шифрования и аутентификации. Радиостанции 802.15.4 предоставляют **список управления доступом (ACL)**

для управления набором и ключами безопасности для использования. Устройства могут хранить до 255 записей ACL.

Уровень MAC также вычисляет «проверки свежести» между последовательными повторениями, чтобы гарантировать, что старые кадры или старые данные больше не считаются действительными и не позволяют этим кадрам переходить в стек.

Каждый трансивер 802.15.4 должен управлять своим ACL и заполнять его списком «доверенных соседей» вместе с политиками безопасности. ACL включает адрес чистого для связи узла с конкретным набором правил безопасности для использования (AES-CTR, AES-CCM-xx, AES-CBCMAC-xx), ключ для алгоритма AES и последний начальный вектор (IV) и *счетчик повторов*. В табл. 5.4 перечислены различные режимы и функции безопасности 802.15.4.

Таблица 5.4. Режимы и функции безопасности в сети типа 802.15.4

Тип	Описание	Контроль доступа	Конфиденциальность	Целостность кадра	Свежесть последовательности
Нет	Безопасность отсутствует				
AES-CTR	Только шифрование, CTR	X	X		X
AES-CBC-MAC-128	128-битный MAC	X		X	
AES-CBC-MAC-64	64-битный MAC	X		X	
AES-CBC-MAC-32	32-битный MAC	X		X	
AES-CCM-128	Шифрование и 128-битный MAC	X	X	X	X
AES-CCM-64	Шифрование и 64-битный MAC	X	X	X	X
AES-CCM-32	Шифрование и 32-битный MAC	X	X	X	X

Симметричная криптография опирается на обе конечные точки, используя один и тот же ключ. Ключами можно управлять на сетевом уровне с помощью общего сетевого ключа. Это простой подход, когда все узлы обладают одним и тем же ключом, но подвержены риску внутрисетевых атак. Можно использовать схему парных ключей, при которой уникальные ключи распределяются между каждой парой узлов. Этот режим добавляет накладные расходы, особенно для сетей, где есть высокий поток от узлов к соседям. Групповые ключи – еще один вариант. В этом режиме один ключ используется совместно набором узлов и используется для любых двух узлов группы. Группы основаны на сходстве устройств, географии и т. д. Наконец, возможен гибридный подход, сочетающий любую из трех упомянутых схем.

Zigbee

Zigbee – это протокол WPAN, основанный на IEEE 802.15.4, предназначенный для коммерческих и жилых сетей IoT, которые ограничены стоимостью, мощностью и пространством. В этом разделе подробно описывается протокол Zigbee с точки зрения аппаратного и программного обеспечения. Zigbee полу-

чил свое название от концепции пчелиного полета. Когда пчела летит туда-сюда между цветами, собирая пыльцу, она напоминает пакет, проходящий через mesh-сеть от устройства к устройству.

История Zigbee

Концепция маломощных сетей беспроводной сети стала стандартной в 1990-х гг., и Альянс Zigbee был сформирован для решения этого регламента в 2002 г. Протокол Zigbee был задуман после ратификации IEEE 802.15.4 в 2004 г. Это стало IEEE 802.15.4.-2003 с 14 декабря 2004 г. Спецификация 1.0, также известная как спецификация Zigbee 2004, была обнародована 13 июня 2005 г. История может быть профилирована как:

- 2005 – выпущен Zigbee 2004;
- 2006 – выпущен Zigbee 2006;
- 2007 – выпущен Zigbee 2007, также известный как ZigbeePro (введенные библиотеки кластеров, некоторые ограничения обратной совместимости с Zigbee 2004 и 2006).

Отношение альянса к рабочей группе IEEE 802.15.4 аналогично рабочей группе IEEE 802.11 и Wi-Fi Alliance. Альянс Zigbee поддерживает и публикует стандарты для протокола, организует рабочие группы и управляет списком профилей приложений. IEEE 802.15.4 определяет уровни PHY и MAC, но ничего выше. Кроме того, в 802.15.4 ничего не говорится о многоходовой связи или пространстве прикладных программ. Именно здесь входит в игру Zigbee (и другие стандарты, основанные на 802.15.4).

Zigbee является частным и закрытым стандартом. Это требует лицензионного сбора и соглашения, предоставленного Альянсом Zigbee. Лицензирование предоставляет сертификат соответствия Zigbee и логотип. Это гарантирует совместимость с другими устройствами Zigbee.

Обзор Zigbee

Zigbee основан на 802.15.4, кроме сетевых уровней, подобных TCP/IP. Он может создавать сети, обнаруживать устройства, обеспечивать безопасность и управлять сетью. Он не предоставляет услуги передачи данных или среду выполнения приложений. Поскольку он по существу является mesh-сетью, то является самовосстанавливающимся и самодостаточным. Кроме того, Zigbee гордится своей простотой и утверждает, что есть 50%-ное сокращение поддержки программного обеспечения за счет использования легкого стека протокола.

В сети Zigbee есть три основных компонента:

- **контроллер Zigbee (ZC)** – высокопроизводительное устройство в сети Zigbee, которое используется для формирования и запуска сетевых функций. Каждая сеть Zigbee будет иметь один ZC, который выполняет роль координатора PAN 802.15.4 2003 PF (FFD). После формирования сети ZC может вести себя как ZR (маршрутизатор Zigbee). Он может назначать логические сетевые адреса и разрешать узлам присоединяться или покидать mesh-сеть;

- **маршрутизатор Zigbee (ZR)** – этот компонент является необязательным, но выполняет некоторую нагрузку на сетевую скачкообразную перестройку и координацию маршрутизации. Он также может выполнять роль FFD и имеет связь с ZC. ZR участвует в маршрутизации сообщений с несколькими переходами и может назначать логические сетевые адреса и разрешать узлам присоединяться или покидать mesh-сеть;
- **конечное устройство Zigbee (ZED)** – обычно это простое оконечное устройство, например, выключатель света или термостат. Оно имеет достаточно функциональных возможностей для общения с координатором. Оно не имеет логики маршрутизации; поэтому любые сообщения, поступающие на ZED, которые не нацелены на это конечное устройство, просто передаются. Оно также не может выполнять ассоциации (подробности позже).

Zigbee нацелен на три разных типа трафика данных. Периодические данные доставляются или передаются со скоростью, определенной приложениями (например, периодически передают датчики). Периодические данные возникают, когда приложение или внешний стимул происходит со случайной задержкой. Хорошим примером прерывистых данных, подходящих для Zigbee, является переключатель освещения. Конечным типом трафика Zigbee служит повторяющиеся данные с низкой задержкой. Zigbee выделяет временные интервалы для передачи и может иметь очень низкую задержку, которая подходит для компьютерной мыши или клавиатуры.

Zigbee поддерживает три основные топологии (рис. 5.22):

- **звездная сеть** – один ZC с одним или несколькими ZED. Только расширяет связь между двумя узлами и поэтому ограничено на расстоянии узла. Также требуется надежная связь с единой точкой отказа в ZC;
- **кластерное дерево** – сеть с несколькими переходами, которая использует маяк и расширяет охват сети и диапазон по сети звезд. Узлы ZC и ZR могут иметь дочерние элементы, но ZED остаются истинными конечными точками. Узлы-потомки взаимодействуют только со своим родителем (например, с небольшой сетью звезд). Родители могут общаться вниз по течению со своими детьми или вверх по потоку до своего родителя. Все еще существует проблема в единой точке отказа в центре;
- **mesh-сеть** – динамическое формирование пути и изменение формы. Маршрутизация может происходить с любого исходного устройства на любое целевое устройство. Использует алгоритмы маршрутизации дерева и таблицы. Радиостанции ZC и ZR должны постоянно быть запитанными, чтобы выполнять требования к маршрутизации, забирая для этого время автономной работы. Кроме того, вычисление задержки в mesh-сети может быть трудным, если не недетерминированным. Некоторые правила ослаблены; однако маршрутизаторы в определенном радиусе друг от друга могут напрямую связываться друг с другом. Основным преимуществом является то, что сеть может вырасти за пределы видимости и иметь несколько добавочных путей.

Zigbee, теоретически, может развернуть до 65536 **конечных устройств Zigbee (ZED)**.

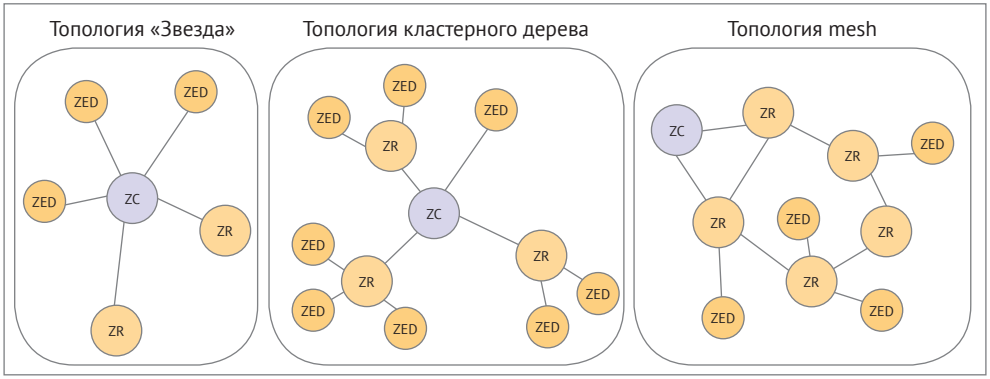


Рис. 5.22 ❖ Три формы топологии сети Zigbee
от простейшей звездной сети и дерева кластера до истинной mesh-сети

Zigbee PHY и MAC (и отличие от IEEE 802.15.4)

Zigbee, как и Bluetooth, работает, в основном, в диапазоне 2,4 ГГц ISM. В отличие от Bluetooth он также работает на частоте 868 МГц в Европе и 915 МГц в США и Австралии. Из-за более низкой частоты он имеет лучшую способность проникать сквозь стены и препятствия по сравнению с традиционными сигналами 2,4 ГГц. Zigbee не использует все спецификации PHY и MAC IEEE 802.15.4. Zigbee использует схему предотвращения столкновений CSMA/CA. Он также использует механизм уровня MAC, чтобы препятствовать тому, чтобы узлы общались друг с другом.

Zigbee не использует режимы маяка IEEE 802.15.4. Кроме того, гарантированные временные интервалы (GTS) SuperFrames также не используются в Zigbee.

Спецификация безопасности 802.15.4 слегка изменена. Режимы CCM, обеспечивающие аутентификацию и шифрование, требуют различной безопасности для каждого уровня. Zigbee ориентирован на сильно ресурсоемкие и глубоко внедренные системы и не обеспечивает уровень безопасности, как определено в 802.15.4.

Zigbee основан на спецификации IEEE 802.15.4-2003, прежде чем усовершенствования с двумя новыми PHY и радиостанциями были стандартизованы в спецификации IEEE 802.15.4-2006. Это означает, что скорость передачи данных несколько ниже, чем в диапазонах 868 МГц и 900 МГц.

Стек протокола Zigbee

Стек протокола Zigbee включает в себя **сетевой уровень (NWK)** и **прикладной уровень (APS)**. Дополнительные компоненты включают поставщика услуг

безопасности, плоскость управления ZDO и **объект устройства Zigbee (ZDO)**. Структура стека иллюстрирует подлинную простоту в отношении более сложного, но функционального пакета Bluetooth (см. рис. 5.23).

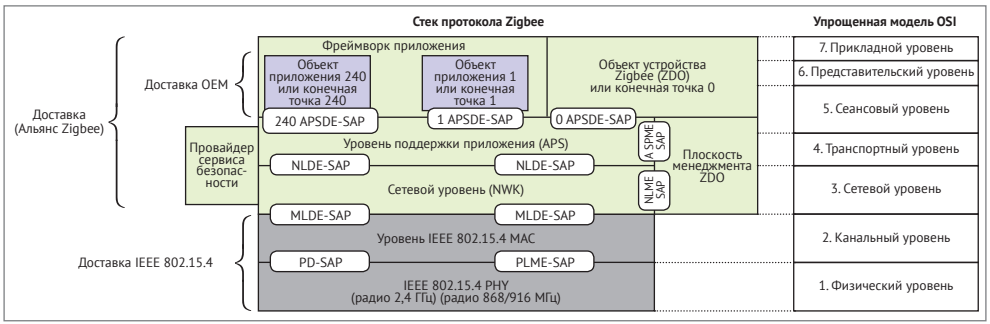


Рис. 5.23 ❖ Стек протокола Zigbee и соответствующая упрощенная модель OSI в качестве системы координат

NWK используется для всех трех основных компонентов Zigbee (ZR, ZC, ZED). Этот уровень выполняет управлением устройством и обнаружением маршрута. Кроме того, поскольку он управляет реальной динамической mesh-сетью, он также отвечает за обслуживание и восстановление маршрута. В качестве самой основной функции NWK отвечает за передачу сетевых пакетов и маршрутизацию сообщений. Во время процесса соединения узлов с mesh-сетью Zigbee NWK обеспечивает логический сетевой адрес для ZC и обеспечивает соединение.

APS обеспечивает интерфейс между сетевым уровнем и уровнем приложения. Он управляет базой данных таблицы привязки, которая используется для поиска правильного устройства в зависимости от требуемой службы и предлагаемой услуги. Приложения моделируются тем, что называют *объектами приложения*. Объекты приложения обмениваются данными друг с другом через карту атрибутов объекта, называемых *кластерами*. Связь между объектами создается в сжатом файле XML, чтобы обеспечить общедоступность. Все устройства должны поддерживать базовый набор методов; однако может существовать в общей сложности 240 конечных точек на устройство Zigbee.

APS связывает устройство Zigbee с пользователем. Здесь присутствуют большинство компонентов протокола Zigbee, включая *объект устройства Zigbee* (ZDO). Конечная точка 0 называется ZDO и является критическим компонентом, который отвечает за общее управление устройствами. Это включает в себя управление ключами, политиками и ролями устройств. Он также может обнаруживать новые (в пределах одного прыжка) устройства в сети и сервисы, предлагаемые этими устройствами. ZDO инициирует и отвечает на все запросы на привязку к устройству. Он также устанавливает безопасную связь между сетевыми устройствами путем управления политикой безопасности и ключами для устройства.

Привязка – это соединения между двумя конечными точками с каждой привязкой, поддерживающей конкретный профиль приложения. Поэтому, когда мы объединяем источник и конечную точку назначения, идентификатор кластера и идентификатор профиля, мы можем создать уникальное сообщение между двумя конечными точками и двумя устройствами. Привязки могут быть индивидуальными, один-ко-многим или многие-к-одному. Примером такого связывания в действии было бы несколько переключателей света, связанных с набором лампочек. Конечные точки приложения коммутатора будут ассоциированы со световыми конечными точками. ZDO обеспечивает управление привязкой, связывая конечные точки коммутатора с легкими конечными точками, используя объекты приложения. Кластеры могут быть созданы, позволяя одному коммутатору включать все огни, в то время как другой переключатель может управлять только одной лампой.

Профиль приложения, поставляемый OEM, будет описывать набор устройств для определенных функций (например, выключатели света и дымовая сигнализация). Устройства в профиле приложения могут взаимодействовать друг с другом через кластеры. Каждый кластер будет иметь уникальный идентификатор кластера, чтобы идентифицировать себя внутри сети.

Адресация и структура пакетов Zigbee

Протокол Zigbee, как показано, находится поверх уровней PHY и MAC 802.15.4 и повторно использует структуры пакетов. Сеть расходится на сетевом и прикладном уровнях.

На рис. 5.24 показано разбиение одного из пакетов PHY и MAC 802.15.4 в соответствующий пакет кадров сетевого уровня (NWK), а также кадр прикладного уровня (APS).

Zigbee использует два уникальных адреса для каждого узла:

- **длинный адрес (64 бит)** – назначается изготовителем устройства и неизменен. Уникально отличает устройство Zigbee от всех других устройств Zigbee. Это то же самое, что и 64-разрядный адрес 802.15.4. Верхние 24 бита относятся к организационному уникальному идентификатору (OUI), а нижние 40 бит управляются OEM. Адреса поставляются блоками и могут быть заказаны в IEEE;
- **короткий адрес (16 бит)** – это тоже то же самое, что и идентификатор PAN для спецификации 802.15.4, является необязательным.

Маршрутизация в mesh-сети Zigbee

Маршрутизация таблиц использует **AdHoc векторную маршрутизацию по требованию (AODV)** и алгоритм дерева кластеров. AODV – это чистая система маршрутизации по требованию. В этой модели узлам не нужно обнаруживать друг друга до тех пор, пока между ними не будет некоторой связи (например, двум узлам необходимо обмениваться данными). AODV также не требует узлов, которые не находятся в пути маршрутизации для поддержания информации маршрутизации. Если исходный узел должен связаться с получателем,

а путь не существует, начинается процесс обнаружения пути. AODV обеспечивает поддержку одноадресной и многоадресной рассылки. Это реактивный протокол; то есть он обеспечивает маршрут к назначению по требованию, а не проактивно. Вся сеть молчит до тех пор, пока не потребуется соединение.

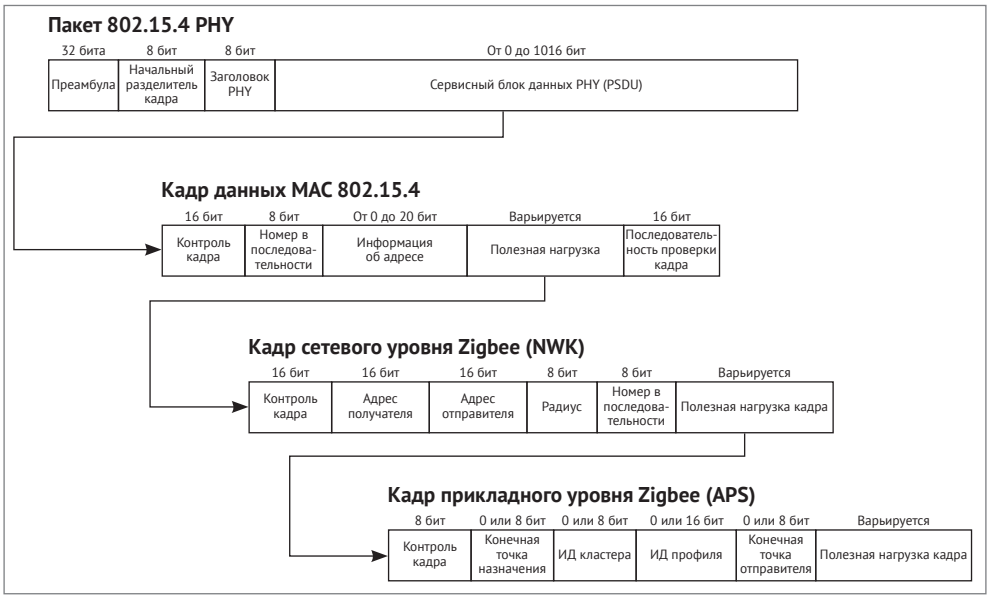


Рис. 5.24 ❖ Сетевой (NWK) и прикладной (APS) уровень Zigbee, которые хранятся в пакетах PHY и MAC 802.15.4

Алгоритм дерева кластеров формирует самоорганизующуюся сеть, способную к самовосстановлению и избыточности. Узлы в mesh-сети выбирают голову кластера и создают кластеры вокруг головного узла. Эти самообразующиеся кластеры затем соединяются друг с другом через *назначенное устройство*.

Zigbee имеет возможность маршрутизировать пакеты несколькими способами:

- **широковещание** – передача пакета ко всем другим узлам в среде;
- **маршрутизация в mesh-сети (табличная маршрутизация)** – если существует таблица маршрутизации для адресата, маршрут будет следовать всем правилам таблицы соответствующим образом. Очень эффективный. Zigbee позволит mesh-сети и таблице маршрутизировать до 30 прыжков;
- **маршрутизация дерева** – одноадресная передача сообщений с одного узла на другой. Маршрутизация дерева является необязательной и может быть запрещена по всей сети. Она обеспечивает лучшую эффективность памяти, чем маршрутизация в mesh-сети, поскольку большая таблица маршрутизации не существует. Однако маршрутизация дерева не

имеет такой же избыточности соединений, как mesh-сети. Zigbee поддерживает древовидную маршрутизацию до 10 узлов;

- **маршрутизация от источника** – используется, главным образом, при наличии концентратора данных. Так Z-Wave обеспечивает маршрутизацию в mesh-сети (см. рис. 5.25).

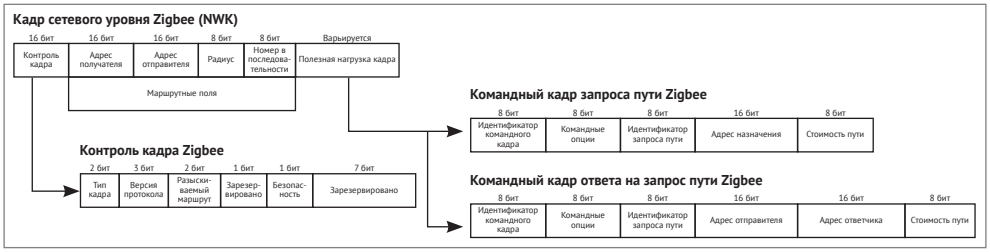


Рис. 5.25 ❖ Zigbee Routing Packet для выдачи командного кадра запроса маршрута и последующего ответа с помощью командного кадра ответа на маршрут

Обнаружение маршрута или обнаружение пути – это процесс обнаружения нового маршрута или восстановления неисправного маршрута. Устройство выдает командный кадр запроса маршрута во всю сеть. Если и когда получатель получает командный кадр, он ответит хотя бы одним командным кадром маршрута. Все возвращенные потенциальные маршруты проверяются и оцениваются, чтобы найти оптимальный маршрут.

Расходы на связь, которые сообщаются при обнаружении пути, могут быть постоянными или основаны на вероятности получения.

Ассоциация Zigbee

Как упоминалось ранее, **конечные устройства Zigbee (ZED)** не участвуют в маршрутизации. Конечные устройства взаимодействуют с родителем, который также является маршрутизатором. Когда **координатор Zigbee (ZC)** позволяет новому устройству присоединиться к сети, он переходит к процессу, известному как *ассоциация*. Если устройство теряет контакт со своим родителем, устройство может в любое время воссоединиться с процессом, известным как *сиротство*.

Чтобы формально присоединиться к сети Zigbee, *запрос маяка* передается устройством для запроса последующих маяков от устройств в mesh-сети, которым разрешено присоединять новые узлы. Сначала только уполномоченный PAN уполномочен предоставлять такой запрос; после того, как сеть выросла, могут участвовать другие устройства.

Безопасность Zigbee

Zigbee строит правила безопасности в соответствии с IEEE 802.15.4. Zigbee предоставляет три механизма безопасности: **списки управления доступом (ACL)**, 128-битное шифрование AES и таймеры свежести сообщений.

Модель безопасности Zigbee распределена между несколькими уровнями:

- уровень приложения обеспечивает создание ключей и транспортные услуги для ZDO;
- сетевой уровень управляет маршрутизацией, а исходящие кадры будут использовать ключ линка, определенный маршрутизацией, если он доступен; в противном случае используется сетевой ключ;
- безопасность уровня MAC управляется через API и контролируется верхними уровнями.

Существует несколько ключей, которыми управляет сеть Zigbee:

- **мастер-ключ** – может быть предварительно установлен производителем или введен вручную пользователем. Он является основой безопасности устройства Zigbee. Мастер-ключи всегда устанавливаются первыми и передаются из центра доверия;
- **сетевой ключ** – этот ключ обеспечит защиту на сетевом уровне от внешних злоумышленников;
- **ключ соединения** – обеспечивает надежное связывание между двумя устройствами. Если у двух устройств есть выбор между использованием установленных ключей связи или сетевых ключей, они всегда будут по умолчанию связывать ключи, чтобы обеспечить дополнительную защиту.

Ключи соединения являются ресурсоемкими в плане хранения ключей на ограниченных устройствах. Сетевые ключи могут использоваться для снижения некоторых из затрат на хранение с учетом снижения безопасности.

Управление ключами имеет решающее значение для безопасности. Распределение ключей контролируется путем создания центра доверия (один узел выступает в роли распределителя ключей ко всем другим узлам в среде). Предполагается, что ZC является центром доверия. Можно реализовать сеть Zigbee с выделенным центром доверия вне ZC. Центр доверия выполняет следующие услуги:

- управление доверием – аутентифицирует устройства, подключающиеся к сети;
- управление сетью – поддерживает и распределяет ключи;
- управление конфигурацией: обеспечивает защиту от устройства до устройства.

Кроме того, центр доверия может быть размещен в режиме резидента (он не будет устанавливать ключи с сетевыми устройствами) или может быть в коммерческом режиме (устанавливает ключи с каждым устройством в сети).

Zigbee использует 128-битные ключи как часть своей спецификации в пределах уровней MAC и NWK. Уровень MAC обеспечивает три режима шифрования: AES-CTR, AES-CBC-128 и AES-CCM-128 (все они определены в разделе IEEE 802.15.4). Однако уровень NWK поддерживает только AES-CCM-128, но слегка улучшен, чтобы обеспечить защиту только шифрования и целостности.

Целостность сообщений гарантирует, что сообщения не были изменены при передаче. Этот тип инструмента безопасности используется для атак типа «человек-посередине». Возвращаясь к структуре пакета Zigbee, код целостности сообщения и вспомогательный заголовок предоставляют поля для добавления дополнительных проверок для каждого отправленного сообщения приложения.

Аутентификация обеспечивается с помощью общего сетевого ключа и отдельных ключей между парами устройств.

Таймеры свежести сообщений используются для поиска сообщений с таймаутом. Эти сообщения отклоняются и удаляются из сети в качестве инструмента для управления атаками повтора. Это относится к входящим и исходящим сообщениям. Каждый раз, когда создается новый ключ, таймеры свежести сбрасываются.

Z-Wave

Z-Wave – это протокол WPAN, используемый прежде всего для потребительской и домашней автоматизации, и около 2100 продуктов используют эту технологию. Он нашел свой путь в коммерческих и строительных сегментах, в областях освещения и управления HVAC. Что касается доли рынка, Z-Wave не настолько популярен, как Bluetooth или Zigbee. Z-Wave – еще одна mesh-технология в диапазоне 900 МГц. Его первое появление было в 2001 г. в Zensys, датской компании, разрабатывающей системы управления освещением. Zensys заключил союз с Leviton Manufacturing, Danfoss и Ingersoll-Rand в 2005 г., официально известный как Z-Wave Alliance. Альянс приобрел Sigma Designs в 2008 г., а Sigma теперь является единственным поставщиком аппаратных модулей Z-Wave.

Компании-члены Z-Wave Alliance теперь включают SmartThings, Honeywell, Belkin, Bosch, Carrier, ADT и LG.

Z-Wave является закрытым протоколом в большинстве случаев с ограниченным количеством производителей аппаратных модулей. Спецификация начинает открываться больше, но существенное количество материала не раскрывается.

Обзор Z-Wave

Дизайн Z-Wave – это домашнее и потребительское освещение/автоматизация. Он предназначен для использования с очень низкой полосой пропускания для связи с датчиками и переключателями. Проект основан на стандарте ITU-T G.9959 на уровне PHY и MAC. ITU-T G.9959 является спецификацией Международного союза электросвязи для короткодействующих узкополосных радиокоммуникационных приемопередатчиков в полосе частот ниже 1 ГГц.

В диапазоне частот до 1 ГГц имеется несколько поддиапазонов, которые используются для Z-Wave в зависимости от страны происхождения. В США центральная частота 908,40 МГц является стандартной. Существуют три скорости передачи данных, которые Z-Wave может использовать с разным распределением частоты для каждого:

- 100 Кбит/с – 916,0 МГц с разбросом 400 кГц;
- 40 Кбит/с – 916,0 МГц с разбросом 300 КГц;
- 9,6 Кбит/с – 908,4 МГц с разбросом 300 КГц.

Каждая полоса работает на одном канале.

Модуляция, выполняемая на уровне PHY, использует частотную манипуляцию для скорости передачи данных 9,6 Кбит/с и 40 Кбит/с. При скорости 100 Кбит/с используется гауссовская манипуляция с частотным переключением. Выходная мощность будет составлять примерно 1 мВт при 0 дБ.

Конфликт каналов управляется с использованием CSMA/CA, как описано в других ранее рассмотренных протоколах. Он управляется на уровне MAC стека. Узлы запускаются в режиме приема и ждут период времени перед передачей данных, если есть данные для широковещания.

С точки зрения роли и ответственности сеть Z-Wave состоит из разных узлов со специфическими функциями:

- **контроллер** – это устройство верхнего уровня обеспечивает таблицу маршрутизации для mesh-сети и является хостом/мастером mesh-сети. Существует два основных типа контроллеров:
  - **главный контроллер** – основной контроллер является ведущим, и только один мастер может существовать в сети. Он имеет возможность поддерживать топологию и иерархию сети. Он также может включать или исключать узлы из топологии. Он также обязан выделять идентификаторы узлов;
  - **вторичный контроллер** – эти узлы помогают первичному контроллеру с маршрутизацией;
- **ведомое устройство/узел** – эти устройства выполняют действия на основе команд, которые они получают. Эти устройства не могут связываться с соседними подчиненными узлами, за исключением случаев, когда это предусмотрено инструкцией с помощью команды. Ведомые устройства могут хранить информацию о маршрутизации, но не вычислять или обновлять таблицы маршрутизации. Как правило, они будут выступать в качестве ретранслятора в mesh-сети.

Контроллеры также могут быть определены как переносные и статические. Портативный контроллер предназначен для перемещения, как пульт дистанционного управления. Как только он изменит положение, он пересчитает самые быстрые маршруты в сети. Статический контроллер предназначен для установки, например, шлюза, подключенного к сетевой розетке. Статический контроллер всегда может быть включен и получать сообщения состояния подчиненного устройства.

Контроллеры также могут иметь разные атрибуты в сети:

- **контроллер обновления состояния (SUC)** – статический контроллер также имеет преимущество в роли контроллера обновления состояния. В этом случае он будет получать уведомления от основного контроллера относительно изменений топологии. Он также может помочь в маршрутизации ведомых;

- **SUC ID-сервер (SIS)** – SUC также может помочь включить и исключить подчиненные устройства для основного;
- **контроллер моста** – это, по сути, статический контроллер, который может действовать как шлюз между сетью Z-Wave и другими сетевыми системами (например, WAN или Wi-Fi). Мосту разрешено управлять до 128 виртуальными подчиненными узлами;
- **контроллер установщика** – это портативный контроллер, который может помочь в управлении сетью и анализе качества обслуживания.

Ведомые также поддерживают разные атрибуты:

- **ведомое устройство маршрутизации** – в основном, подчиненный узел, но с возможностью отправки незапрашиваемых сообщений другим узлам в mesh-сети. Как правило, ведомым устройствам запрещено отправлять сообщение другому узлу без команды основного контроллера. Узел хранит набор статических маршрутов, которые он использует при отправке сообщения;
- **расширенное ведомое устройство** – они обладают теми же способностями, что и ведомое устройство маршрутизации, с добавлением часов реального времени и постоянной памяти для данных приложения. Примером может служить газовый счетчик;

Ведомые узлы/устройства могут работать на батарее, например, датчик движения в доме. Для того, чтобы подчиненный был ретранслятором, он всегда должен функционировать и прослушивать сообщения в mesh-сети. Из-за этого устройства на батарейках никогда не используются в качестве повторителей.

Стек протокола Z-Wave

Поскольку Z-Wave является протоколом с очень низкой пропускной способностью, который предназначен для разреженной топологии сети, стек протокола пытается выделить как можно меньше байтов на сообщение. Стек состоит из пяти уровней, как показано на рис. 5.26.

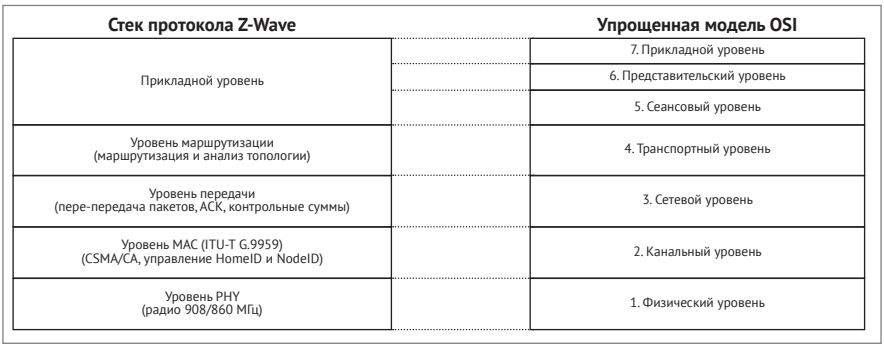


Рис. 5.26 ❖ Сравнение стека протоколов Z-Wave и модели OSI. Z-Wave использует пятиуровневый стек с нижними двумя уровнями (PHY и MAC), определенными спецификацией ITU-T G.9959

Уровни могут быть описаны следующим образом:

- **PHY-уровень** – определяется спецификацией ITU-T G.9959. Этот уровень управляет модуляцией сигнала, назначением канала и привязкой преамбулы к передатчику и синхронизацией преамбулы в приемнике;
- **уровень MAC** – этот уровень управляет полями HomeID и NodeID, как определено в предыдущем разделе. Уровень MAC также использует алгоритм предотвращения столкновений и стратегию отсрочки, чтобы уменьшить перегрузку и конфликты на канале;
- **уровень трансфера** – управляет обменом кадрами Z-Wave. Этот уровень также отвечает за повторную передачу кадров по мере необходимости. Дополнительные задачи включают подтверждение передачи и привязки контрольной суммы;
- **уровень маршрутизации** – предоставляет услуги маршрутизации. Кроме того, сетевой уровень будет выполнять сканирование топологии и обновление таблиц маршрутизации;
- **уровень приложения** – предоставляет пользовательский интерфейс приложениям и данным.

Адресация Z-Wave

Z-Wave имеет довольно простой механизм адресации по сравнению с протоколами Bluetooth и Zigbee. Схема адресации остается простой, поскольку предпринимаются все попытки минимизировать трафик и сэкономить электроэнергию. Существует два основных идентификатора адресации, которые требуют определения:

- **домашний идентификатор** – это 32-битный уникальный идентификатор, который предварительно запрограммирован в контроллерах, чтобы помочь идентифицировать сети Z-Wave друг от друга. Во время запуска сети все ведомые устройства Z-Wave имеют домашний идентификатор 0, и контроллер будет систематически заполнять подчиненные узлы правильным домашним идентификатором;
- **идентификатор узла** – это 8-битное значение, которое назначается каждому подчиненному контроллеру и обеспечивает адресацию ведомых устройств в сети Z-Wave (рис. 5.27).

Транспортный уровень предоставляет несколько типов кадров для оказания помощи при повторной передаче, подтверждении, управлении мощностью и аутентификации. Четыре типа сетевых кадров включают:

- **кадр одноадресной передачи** – это пакет, отправленный на один узел Z-Wave. За этим типом пакета должно следовать подтверждение. Если ACK нет, возникает последовательность повторной передачи;
- **кадр ACK** – это ответ подтверждения для кадра одноадресной передачи;
- **многоадресный кадр** – это сообщение передается на несколько узлов (до 232). Для этого типа сообщений не используется подтверждение;
- **кадр широковещательной передачи** – подобно многоадресному сообщению, этот кадр передается всем узлам сети. ACK не используется.

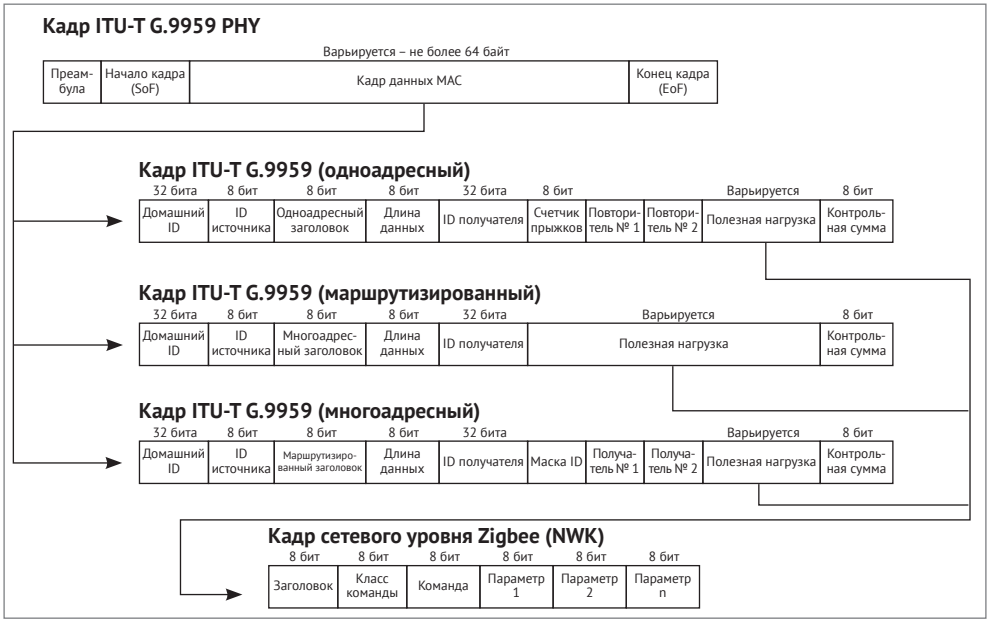


Рис. 5.27 ❖ Структура пакета Z-Wave от PHY до MAC на прикладном уровне. Также определены три типа пакетов: одноадресная передача, маршрутизация и многоадресная рассылка

Для нового устройства Z-Wave, которое будет использоваться в mesh-сети должен пройти процесс сопряжения и добавления. Процесс обычно запускается механическим или пользовательским нажатием на устройстве. Как уже упоминалось, процесс сопряжения включает в себя первичный контроллер, назначающий домашний идентификатор новому узлу. На этом этапе считается, что узел включен.

Топология и маршрутизация Z-Wave

Топология mesh-сети Z-Wave показана на рис. 5.28, используя некоторые типы устройств и атрибуты, связанные с подчиненными устройствами и контроллерами. Один первичный контроллер управляет сетью и устанавливает поведение маршрутизации.

Уровень маршрутизации в стеке Z-Wave управляет транспортировкой кадров с одного узла на другой. Уровень маршрутизации настраивает правильный список ретрансляторов, если он необходим, сканирует сеть для изменений в топологии и поддерживает таблицу маршрутизации. Таблица довольно проста и указывает, какой сосед подключен к данному узлу. Она отображает только один быстрый прыжок. Таблица построена главным контроллером, задавая каждому узлу в mesh-сети данные, какие устройства доступны из его местоположения.

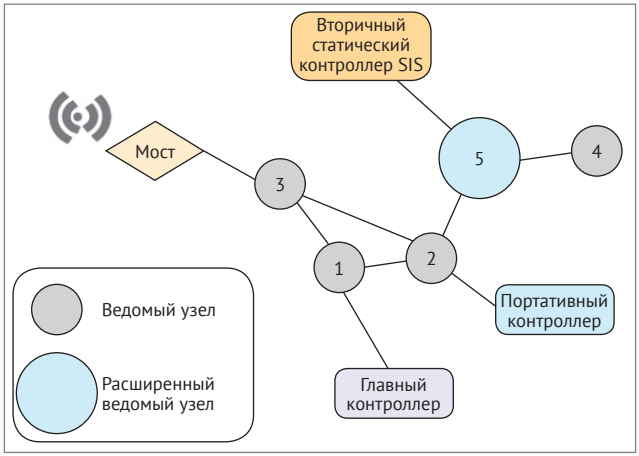


Рис. 5.28 ❖ Z-Wave, включает один основной контроллер и четыре подчиненных устройства и один расширенный ведомый. Контроллер моста выступает в качестве шлюза для сети WiFi. Портативный контроллер и дополнительный контроллер также размещаются в mesh-сети для помощи основному контроллеру

Использование маршрутизации от источника для навигации в mesh-сети подразумевает, что по мере того, как сообщение проходит через среду, на каждом прыжке при получении кадра узел передает пакет следующему узлу в цепочке. Например, в таблице маршрутизации на рис. 5.29 кратчайший путь от «Bridge» до «Slave 5» следует по логическому пути: Bridge | Slave 3 | Slave 2 | Slave 5.

Z-Wave ограничивает переходы маршрутизации максимум до четырех.

Таблица маршрутизации приведенного примера топологии выглядит так:

	Ведомый 1	Ведомый 2	Ведомый 3	Ведомый 4	Расширенный ведомый 5	Основной контроллер	Вторичный SIS	Шлюз	Портативный контроллер
Ведомый 1	0	1	1	0	0	1	0	0	0
Ведомый 2	1	0	1	0	1	0	0	0	1
Ведомый 3	1	1	0	0	0	0	0	1	0
Ведомый 4	0	0	0	0	1	0	0	0	0
Расширенный ведомый 5	0	1	0	1	0	0	1	0	0
Основной контроллер	0	0	0	0	0	0	0	0	0
Вторичный SIS	0	0	0	0	1	0	0	0	0
Шлюз	1	0	1	0	0	0	0	0	0
Портативный контроллер	0	1	0	0	0	0	0	0	0

Рис. 5.29 ❖ Алгоритмический пример маршрутизации от источника Z-Wave

У портативных контроллеров есть проблема в поддержании оптимального маршрута маршрутизации; поэтому переносные контроллеры будут использовать альтернативные методы для поиска наилучшего маршрута к узлу назначения.

Заключение

В этой главе рассмотрен первый шаг в доставке данных IoT с устройств в интернет. Первым шагом в подключении миллиардов устройств является использование правильного средства связи для доступа к датчикам, объектам и исполнительным устройствам, чтобы вызвать некоторые действия. Это роль беспроводной персональной сети. Мы изучили основы нелицензионного спектра и выяснили, как в качестве архитектора мы будем измерять производительность и поведение WPAN. Мы углубились в новый протокол Bluetooth 5 и сравнили его с другими стандартами, такими как базовый протокол IEEE 802.15.4, Zigbee и Z-Wave. Мы изучили маяки, различные структуры пакетов и протоколов, а также архитектуру mesh-сети. На этом этапе архитектор должен понимать, как эти архитектуры сравниваются и контрастируют.

В следующей главе будут рассмотрены IP-сети PAN и LAN, такие как вездесущая сеть Wi-Fi 802.11, Thread и 6LoWPAN, а также будущие стандарты связи Wi-Fi. Эти главы, изучающие архитектуры WPAN, WLAN и WAN, будут регулярно возвращаться к основным принципам, таким как измерения уровня сигнала и уравнения дальности, которые были изложены в начале этой главы, и их следует использовать для справки в будущем.

Глава 6

WPAN и WLAN на базе IP

Сети WPAN взяли протоколы, которые обычно не являются TCP/IP, по крайней мере, с самого начала. Стеки протоколов для Bluetooth, Zigbee и Z-Wave имеют сходство с истинным протоколом TCP/IP, но не общаются по сети через TCP/IP. Существуют адаптации IP на Zigbee (используя Zigbee-IP) и IP через Bluetooth (с использованием IPSP для поддержки 6LoWPAN). Позже в этой главе мы рассмотрим пример WPAN с использованием протокола 802.15.4 с уровнем совместимости с истинным протоколом IPv6 (поток), способным подключаться к любой сети IPv6.

В этой главе также будут рассмотрены стандарты Wi-Fi™, использующие протоколы IEEE 802.11. Как правило, в беспроводной локальной сети (WLAN) протоколы 802.11 широко распространены и используются в реализациях IoT, особенно в интеллектуальных датчиках и концентраторах шлюза. В этой главе будет формально рассмотрен каталог стандартов 802.11 Wi-Fi, включая новый высокоскоростной протокол IEEE 802.11ac. 802.11 также распространяется в мир IoT для транспортного рынка, используя протокол 802.11p, который также будет рассмотрен. Наконец, в этой главе будет рассмотрена спецификация IEEE 802.11ah, которая представляет собой решение для беспроводной локальной сети, основанное на протоколе 802.11, но явно предназначенное для мощных устройств с датчиками и с ограничением затрат.

В главе рассматриваются следующие темы:

- протокол интернета (IP) и протокол управления передачей (TCP);
- WPAN с IP – 6LoWPAN;
- WPAN с IP – поток;
- протоколы IEEE 802.11 и WLAN.

Протокол интернета и протокол управления передачей

Поддержка уровня IP в стеке протоколов потребляет ресурсы, которые могут быть применены в другом месте. Однако есть ключевые преимущества в построении системы IoT, которая позволяет устройствам обмениваться данными через TCP/IP (протокол управления передачей / интернет-протокол). Мы начнем с перечисления этих преимуществ; однако роль архитектора заключается

в том, чтобы сбалансировать стоимость этих услуг и функций и воздействие на систему.

Роль протокола IP в интернете вещей

С точки зрения экосистемы, независимо от протокола, используемого на уровне датчика, данные датчика в конечном итоге будут передаваться в общественное, частное или гибридное облако для анализа, контроля или мониторинга. Вне WPAN, мир основан на TCP/IP, как мы видим в конфигурациях WLAN и WAN.

IP является стандартной формой глобальной связи по различным причинам:

- **повсеместность** – стеки IP предоставляются почти каждой операционной системой и каждой средой. Протоколы IP-связи могут работать в различных системах WPAN, сотовых, проводных, оптоволоконных, PCI Express и спутниковых системах. IP определяет точный формат для всех сообщений и правил, используемых для общения, подтверждения и управления связью;
- **долговечность** – TCP был создан в 1974 г., а стандарт IPv4, который все еще используется сегодня, был разработан в 1978 г. Он выдерживает испытание временем уже 40 лет. Долговечность имеет первостепенное значение для многих промышленных и полевых решений IoT, которые должны поддерживать устройства и системы на протяжении десятилетий. Различные проприетарные протоколы были разработаны различными производителями за эти 40 лет, такие как AppleTalk, SNA, DECnet и Novell IPX, но ни один из них не завоевал рынок так же, как IP;
- **основан на стандартах** – TCP/IP управляется целевой группой Internet Engineering Task Force (IETF). IETF поддерживает набор открытых стандартов, ориентированных на интернет-протокол;
- **масштабируемость** – IP продемонстрировал масштаб и внедрение. IP-сети продемонстрировали значительное масштабирование для миллиардов пользователей и многих других устройств. IPv6 может обеспечить уникальный IP-адрес для каждого атома Земли и поддерживать еще 100 миров;
- **надежность** – IP по своей сути является надежным протоколом для передачи данных. Он выполняет это через систему доставки пакетов на основе сети без установления соединения. Услуга считается ненадежной по концепции, то есть доставка данных не гарантируется. IP является протоколом без установления соединения, поскольку каждый пакет обрабатывается независимо от других. IP также называется *доставкой с наилучшими усилиями*, потому что будут предприняты все попытки передать пакет по различным маршрутам. Сила этой модели позволяет архитектору заменить механизм доставки другим – по существу, заменяя уровни один и два стека чем-то другим (например, Wi-Fi с сотовой);

- **управляемость**: существуют различные инструменты для управления IP-сетями и устройствами в сети IP. Существуют инструменты моделирования, сетевые снифферы, диагностические инструменты и различные устройства, которые помогают создавать, масштабировать и поддерживать сети.

Транспортный слой также стоит рассмотреть. Так как IP требует хорошо поддерживаемого и надежного сетевого уровня, для транспортного уровня необходимы протокол TCP и **Universal Datagram Protocol (UDP)**. Транспортный уровень отвечает за связь из конца в конец. Логическая связь между различными хостами и различными сетевыми компонентами регулируется на этом уровне. TCP используется для соединений, ориентированных на соединение, тогда как UDP используется для передач без установления соединения. Разумеется, UDP намного проще реализовать, чем TCP, но он не настолько устойчив. Обе службы обеспечивают переупорядочение сегментов, поскольку доставка пакетов по порядку не гарантируется с использованием протокола IP. TCP также обеспечивает уровень надежности ненадежного уровня IP-сети за счет использования сообщений подтверждения и повторных передач потерянных сообщений. Кроме того, TCP обеспечивает управление потоком с использованием скользящих окон и алгоритмов предотвращения переполнения. UDP обеспечивает легкий высокоскоростной метод для передачи данных различным устройствам, которые могут существовать или не существовать и быть или не быть надежными.

В табл. 6.1 приведен стандартный 7-уровневый стек модели Open Source Interconnection (OSI).

Таблица 6.1. Стандартный 7-уровневый стек модели Open Source Interconnection (OSI)

Уровень	Цель/функция	Используемые протоколы	Основной тип данных
7. Прикладной	Уровень пользовательских приложений: ftp, app, etc. (удаленный доступ к файлам, разделение ресурсов, LDAP, SNMP)	SMTP, FTP	Данные
6. Представительский	Уровень синтаксиса: шифрование, сжатие (опционально) (шифрование/дешифрование данных, кодеки, перевод)	JPEG, ASCII, ROT 13	Данные
5. Сеансовый	Синхронизация и логическая маршрутизация портов (установление сеанса, старт и окончание безопасности, журналирование, распознавание имен)	RPC, NFS, NetBIOS	Данные
4. Транспортный	TCP: контроль связи хоста с хостом и передвижения данных (соединения из конца в конец, их надежность, сегментация сообщений, подтверждения, мультиплексирование сеанса)	TCP/UDP	TCP: сегменты UDP: датаграммы
3. Сетевой	Пакеты: IP-адрес (определение пути, маршрутизация по логической адресации, контроль трафика, сегментация кадров, управление подсетями)	IP, IPX, ICMP	Пакеты

Таблица 6.1 (окончание)

Уровень	Цель/функция	Используемые протоколы	Основной тип данных
2. Канальный	Кадры данных: MAC-адреса, пакет (физическая адресация Media Access Control, LLC, проверка кадров на ошибки, выставление в очередь и переставление)	PPP/SLIP	Кадры
1. Физический	Физическое устройство: кабели, оптические линии, радиоспектр (кодирование данных, доступность среды передачи, сигнализация о доступной базовой/расширенной полосе передачи, бинарная передача)	Коаксиальный кабель, оптика, радиосвязь	Биты/сигналы

Это полная семиуровневая модель OSI. TCP/IP представляет уровни три и четыре.

С точки зрения IoT, приближение IP-адреса к источнику данных соединяет мостом два мира управления данными. **Информационные технологии (ИТ)** управляют инфраструктурой, безопасностью и обеспечением сетей и объектов в сети. **Операционные технологии (ОТ)** управляют состоянием и пропускной способностью системы, которая функционирует для производства чего-либо. Эти две роли традиционно разделены, поскольку такие вещи, как датчики, счетчики и программируемые контроллеры, не были связаны, по крайней мере, напрямую. Собственные стандарты регулируют системы ОТ, по крайней мере, с точки зрения промышленного IoT.

WPAN с IP – 6LoWPAN

В целях обеспечения адресации IP для самых маленьких и наиболее ограниченных по ресурсам устройств в 2005 г. была сформирована концепция 6LoWPAN. Рабочая группа формализовала дизайн в IETF в соответствии с спецификацией RFC 4944 (запрос для комментариев), а затем обновилась с RFC 6282 до сжатия заголовка адреса и RFC 6775 для обнаружения соседей. Консорциум закрыт, однако стандарт открыт для всех, кто его использует и реализует. 6LoWPAN – это аббревиатура, обозначающая IPV6 с использованием низкомощных WPAN. Цель состоит в том, чтобы использовать IP-сети через системы связи RF с низким энергопотреблением для устройств, которые ограничены в мощности и пространстве и не нуждаются в высокоскоростных сетевых сервисах. Протокол может использоваться с другими протоколами WPAN, такими как 802.15.4, а также с Bluetooth, радиочастотными протоколами 1 ГГц и **контроллером электросети (PLC)**. Основным преимуществом 6LoWPAN является то, что простейший из датчиков может иметь IP-адресность и действовать как участник сети через маршрутизаторы 3G/4G/LTE/Wi-Fi/Ethernet. Дополнительным эффектом является то, что IPV6 обеспечивает значительную теоретическую адресность 2^{128} или $3{,}4 \times 10^{38}$ уникальных адресов. Это позволит в достаточной степени охватить потенциально 50 миллиардов подключенных к интернету устройств к 2020 г. и будет покрывать те устройства, которые находятся за пределами этого. Поэтому IPV6 хорошо подходит для развития интернета вещей.

Топология 6LoWPAN

Сети 6LoWPAN являются смешанными (mesh-) сетями, расположенными на периферии больших сетей. Топологии гибкие, что позволяет создавать специальные и несвязанные сети без привязки к интернету или другим системам, или же они могут быть подключены к магистрали или интернету с использованием граничных маршрутизаторов. Сети 6LoWPAN могут быть объединены с несколькими пограничными маршрутизаторами – это называется **многоточечным подключением**. Кроме того, специальные сети могут формироваться без необходимости подключения к интернету пограничного маршрутизатора. Эти топологии показаны на рис. 6.1.

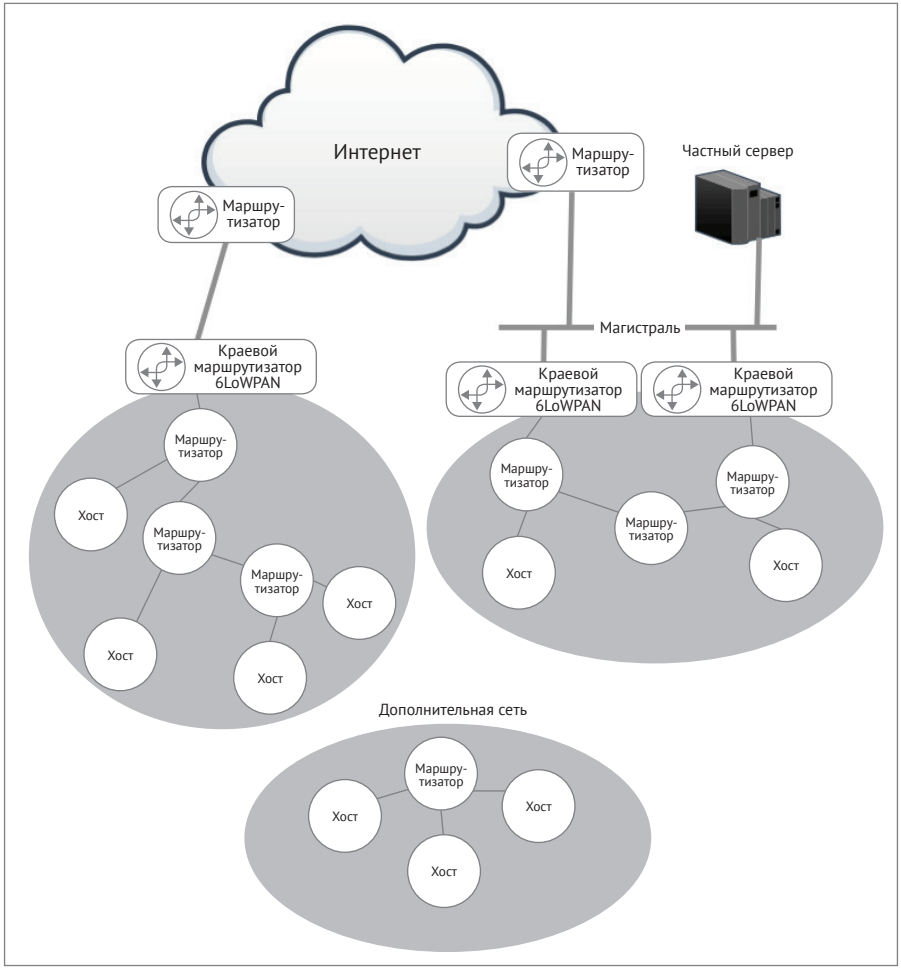


Рис. 6.1 ❖ Топологии 6LoWPAN

Граничный маршрутизатор (также известный как краевой или пограничный маршрутизатор) необходим для архитектуры 6LoWPAN, поскольку он имеет четыре функции:

- поддерживает связь с устройствами 6LoWPAN и передает данные в интернет:
- выполняет сжатие заголовков IPv6, уменьшая 40-байтовый заголовок IPv6 и 8-байтные UDP-заголовки для повышения эффективности в сети датчиков. Типичный 40-байтовый заголовок IPv6 может сжиматься до 2–20 байт в зависимости от использования;
- инициирует сеть 6LoWPAN;
- занимается обменом данных между устройствами в сети 6LoWPAN.

Граничные маршрутизаторы образуют mesh-сети 6LoWPAN на более крупных традиционных сетевых периметрах. Они также могут осуществлять обмен между IPV6 и IPV4, если это необходимо. Дейтаграммы обрабатываются так же, как в IP-сети, что имеет некоторые преимущества по сравнению с проприетарными протоколами. Все узлы в сети 6LoWPAN имеют один и тот же префикс IPv6, который устанавливает пограничный маршрутизатор. Узлы будут регистрироваться на пограничных маршрутизаторах как часть фазы **обнаружения сети (ND)**.

ND определяет, как хосты и маршрутизаторы в локальной сети 6LoWPAN будут взаимодействовать друг с другом. Многоточечность позволяет нескольким пограничным маршрутизаторам 6LoWPAN управлять сетью; например, когда для отказоустойчивости или устойчивости к сбоям требуется наличие нескольких носителей (4G и Wi-Fi).

Существует три типа узлов в mesh-сети 6LoWPAN:

- **узлы маршрутизатора –** эти узлы направляют данные с одного узла сетки 6LoWPAN на другой. Маршрутизаторы также могут передавать информацию в WAN и интернет;
- **узлы хоста –** хосты в mesh-сети не могут маршрутизировать данные в сети и являются просто конечными точками, потребляющими или создающими данные. Хостам разрешено находиться в состоянии сна и иногда пробуждаться для получения данных самим или получения данных, кэшированных их родительскими маршрутизаторами;
- **граничные маршрутизаторы**: как указано, это шлюзы и контроллеры сетки, обычно на краю WAN. Mesh-сеть 6LoWPAN будет управляться граничным маршрутизатором.

Узлы могут свободно перемещаться и реорганизовываться/собираться в сети. В этом случае узел может перемещаться и ассоциироваться с другим пограничным маршрутизатором в сценарии с несколькими управляющими элементами или даже перемещаться между различными сетями 6LoWPAN. Эти изменения в топологии могут быть вызваны разными причинами, такими как изменение уровня сигнала или физическое перемещение узлов. При изменении топологии адрес IPv6 связанных узлов также естественно изменяется.

В специальной сетке без граничного маршрутизатора узел маршрутизатора 6LoWPAN мог бы управлять сеткой 6LoWPAN. Это тот случай, когда подключение WAN к интернету не требуется. Как правило, это редко встречается, поскольку адресация IPv6 для небольшой специальной сети не требуется. Узел маршрутизатора будет настроен для поддержки двух обязательных функций:

- создание уникального локального адреса;
- выполнение регистрации соседа ND.

Префикс добавленной mesh-сети IPv6 был бы локальным префиксом, а не более крупным глобальным префиксом WAN IPv6.

Стек протокола 6LoWPAN

Чтобы использовать 6LoWPAN как коммуникационную среду, такую как 802.15.4, существует набор рекомендуемых функций, необходимых для поддержки IP-протокола. Эти функции включают в себя кадрирование, одноадресную передачу и адресацию. Как показано на рис. 6.2, физический уровень отвечает за прием и преобразование бит данных по воздуху. Примером того, о чем мы говорим, является IEEE 802.15.4. В дополнение к физическому уровню находится канальный уровень, отвечающий за обнаружение и исправление ошибок на физическом уровне. Подробная информация о физическом и канальном уровнях стандарта 802.15.4 приведена ранее в этой главе. Вот сравнение между стеком 6LoWPAN и моделью OSI:

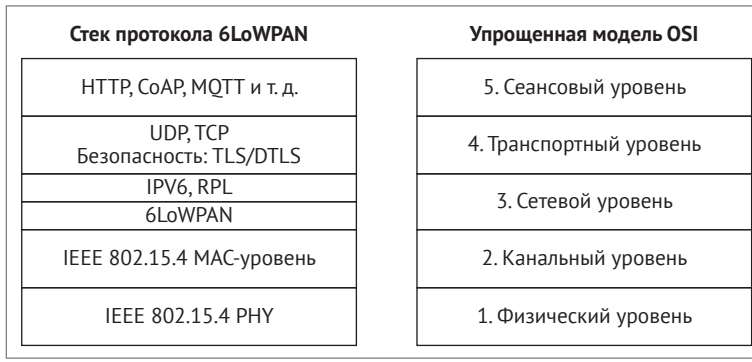


Рис. 6.2 ❖ Сравнение LoWPAN Protocol Stack с упрощенной моделью OSI. 6LoWPAN находится поверх других протоколов, таких как 802.15.4 или Bluetooth, для предоставления физического и MAC-адреса

Включив IP-трафик на уровне датчика, связь между устройством и шлюзом будет использовать некоторый вид прикладного уровня для преобразования данных из протокола, отличного от IP, и IP-протокола. Bluetooth, Zigbee и Z-Wave имеют некоторую форму преобразования от своего базового протокола к тому, что может связываться через IP (если целью является маршрутизация данных). Граничные маршрутизаторы пересылают дейтаграммы на сетевой уровень. Из-за этого маршрутизатору не требуется поддерживать со-

стояние приложения. Если изменяется протокол приложения, шлюз 6LoWPAN не заботится об этом. Если протокол приложения изменяется для одного из протоколов, отличных от IP, шлюз также должен будет изменить свою прикладную логику. 6LoWPAN обеспечивает уровень адаптации на уровне три (сетевой уровень) и поверх второго уровня (канальный уровень). Этот уровень адаптации определяется IETF.

Адресация и маршрутизация в mesh-сети

Маршрутизация в mesh-сети (смешанной сети) работает на физическом и канальном уровнях, чтобы позволить пакетам проходить через динамическую сеть с использованием нескольких переходов. Ранее мы говорили о маршрутизации через сеть и маршрутизацию по пути, но в этом разделе мы рассмотрим эту форму маршрутизации.

В mesh-сетях 6LoWPAN используются две схемы маршрутизации:

- **mesh-under-сеть** – в топологии с сетью маршрутизация прозрачна и предполагает одну IP-подсеть, представляющую всю совокупность сети. Сообщение транслируется в одном домене и отправляется на все устройства в сети. Как упоминалось ранее, это создает значительный трафик. Маршрутизация Mesh-переходов будет из прыжка до прыжка в сети, но пересылаютсятолько пакеты до уровня два (канальный уровень) стека. 802.15.4 обрабатывает всю маршрутизацию для каждого прыжка на втором уровне;
- **маршрутная сеть**: в топологии маршрутизации сети будут взимать плату переадресации пакетов до уровня 3 (сетевой уровень) стека. Схемы маршрутизации управляют маршрутами на уровне IP. Каждый прыжок представляет собой один IP-маршрутизатор.

На рис. 6.3 показана разница между mesh-under-сетью и маршрутной сетью.

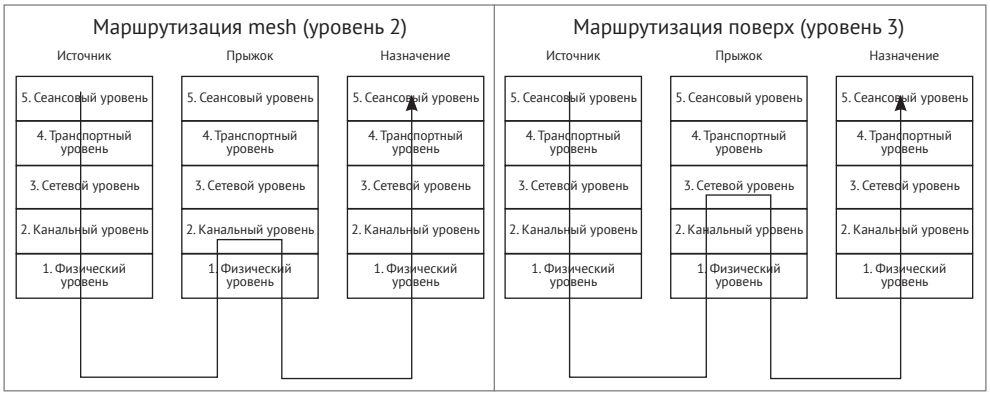


Рис. 6.3 ❖ Разница между mesh-under-сетью и маршрутизируемой сетью. Промежуточные переходы показывают, насколько далеко по каждому стеку доставляется пакет, прежде чем перейти к следующему узлу в сети

Сеть маршрутизации подразумевает, что каждый роутер-узел в равной степени способен и может выполнять больший набор функций в качестве обычного IP-маршрутизатора, например, для обнаружения дублированных адресов. RFC6550 формально определяет протокол маршрутизации RPL (пульсация). Преимуществом архитектуры маршрутизации является сходство с традиционной связью TCP/IP. RPL обеспечивает многоточечную связь (где трафик от устройств в сетке проходит к центральному серверу в интернете) и связь точка-многоточка (центральная служба для устройств в сети).

Протокол RPL имеет два режима управления таблицами маршрутов:

- **режим сохранения** – все устройства, настроенные как маршрутизаторы в сети 6LoWPAN, поддерживают таблицы маршрутизации и соседства;
- **режим без сохранения** – только одно устройство, такое как граничный маршрутизатор, поддерживает таблицы маршрутизации и соседей. Для передачи данных с одного хоста на другой в сетке 6LoWPAN данные отправляются на маршрутизатор, где его маршрут вычисляется, а затем передается в приемник.

Таблица маршрутизации, как следует из названия, содержит пути маршрутизации сети, в то время как таблица соседей поддерживает напрямую связанных соседей каждого узла. Это означает, что граничный маршрутизатор всегда будет отвечать за доставку пакетов в сеть. Это позволяет узлам маршрутизатора свободно управлять большими таблицами маршрутизации для управления, но добавляет некоторую задержку к перемещению пакетов, так как к пограничному маршрутизатору необходимо быть в таблице маршрутизации других маршрутизаторов. Системы хранения данных будут иметь более высокие требования к обработке и памяти для управления таблицами маршрутизации, хранящимися на каждом узле, но будут иметь более эффективный путь для создания маршрута.

На рис. 6.4 обратите внимание на количество точек переходов, адреса источника и адреса приемщика. Эти поля используются во время разрешения адреса и фазы маршрутизации. Счетчик переходов устанавливается на начальное высокое значение и затем уменьшается каждый раз, когда пакет распространяется от узла к узлу в сети. Цель состоит в том, что, когда предел прыжков достигает нуля, пакет отбрасывается и теряется в сети. Это обеспечивает возможность предотвращения безработных сетей в случае, если узел удаляется из сети и больше недоступен. Адрес источника и получателя – это адреса 802.15.4, он может быть коротким или расширенным, как позволяет 802.15.4. На рис. 6.4 показано, как сконструирован заголовок.

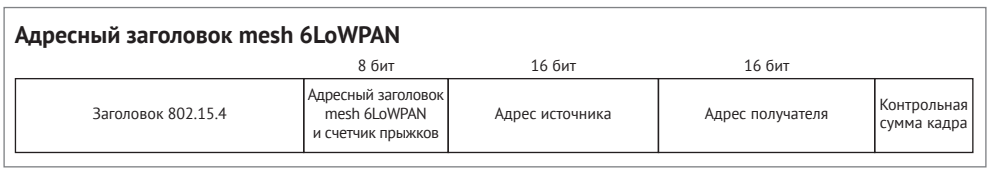


Рис. 6.4 ❖ Заголовок mesh-сети 6LoWPAN

Сжатие и фрагментация заголовка

Хотя преимущество наличия практически неограниченных IP-адресов является важной вехой, помещение IPv6 на линию 802.15.4 создает некоторые проблемы, которые необходимо преодолеть, чтобы использовать 6LoWPAN. Во-первых, IPv6 имеет **максимальный размер блока передачи (MTU)** 1280 байт, в то время как 802.15.4 имеет ограничение в 127 байт. Второй вопрос заключается в том, что IPv6 в целом добавляет значительный объем к уже раздутому протоколу. Например, в заголовках IPv6 длина файла 40 байт.

IEEE 802.15.4g не имеет ограничения на 127 байт для длины кадра.

Сжатие заголовка является средством для сжатия и удаления избыточности в стандартном заголовке IPv6 по соображениям эффективности. Как правило, сжатие заголовков основано на статусе, что означает, что в сети со статическими линками и стабильными соединениями, оно работает достаточно хорошо. В mesh-сети, такой как 6LoWPAN, это не сработает. Пакеты переходят между узлами и требуют сжатия/декомпрессии на каждом прыжке. Кроме того, маршруты являются динамическими и могут меняться, и передачи могут отсутствовать в течение длительного времени. Поэтому 6LoWPAN принял сжатие без сохранения состояния и совместно используемого контекста. Тип сжатия может зависеть от того, выполняются ли определенные спецификации, например, с использованием RFC4944 по RFC6922, а также в зависимости от того, где находятся источник и место назначения пакета, как показано на рис. 6.5.

Три случая сжатия заголовка для 6LoWPAN в зависимости от того, находится ли маршрут в локальной сетке, вне сетки, но с известным адресом или вне сети с неизвестным адресом. По сравнению со стандартным IPv6 с 40-байтным заголовком 6LoWPAN может сжимать в переделах от 2 до 20 байт.

Первый случай (на рис. 6.5) – это наилучшая связь между узлами в локальной сети. С этим сжатым форматом заголовка данные не отправляются наружу в WAN. Второй случай означает, что данные отправляются наружу в WAN по известному адресу, а последний случай аналогичен, но относится к известному адресу. Даже в третьем случае, в худшем случае, сжатие по-прежнему составляет 50%-ное сокращение трафика. 6LoWPAN также допускает сжатие UDP, что выходит за рамки этой книги.

Фрагментация является вторичной проблемой из-за того, что размеры MTU несовместимы между 802.15.4 (127 байт) и IPv6 при 1280 байт. Система фрагментации разделит каждый кадр IPv6 на более мелкие сегменты. На принимающей стороне фрагменты будут повторно собраны. Фрагментация будет различаться в зависимости от типа маршрутизации, выбранного при конфигурировании сети (позже мы обсудим маршрутизацию в сети и пути). Типы фрагментации и ограничений таковы:

- **фрагментация mesh-сети** – фрагменты будут собраны только в конечном расположении. Все фрагменты необходимо учитывать при повтор-

ной сборке. Если они отсутствуют, весь пакет требует повторной передачи. Попутно отметим, что системы со mesh-сетью требуют, чтобы все фрагменты были переданы немедленно. Это создаст пик трафика;

- **фрагментация по пути маршрутизации** – фрагменты будут собираться на каждом переходе в сети. Каждый узел по маршруту имеет достаточные ресурсы и информацию для восстановления всех фрагментов.

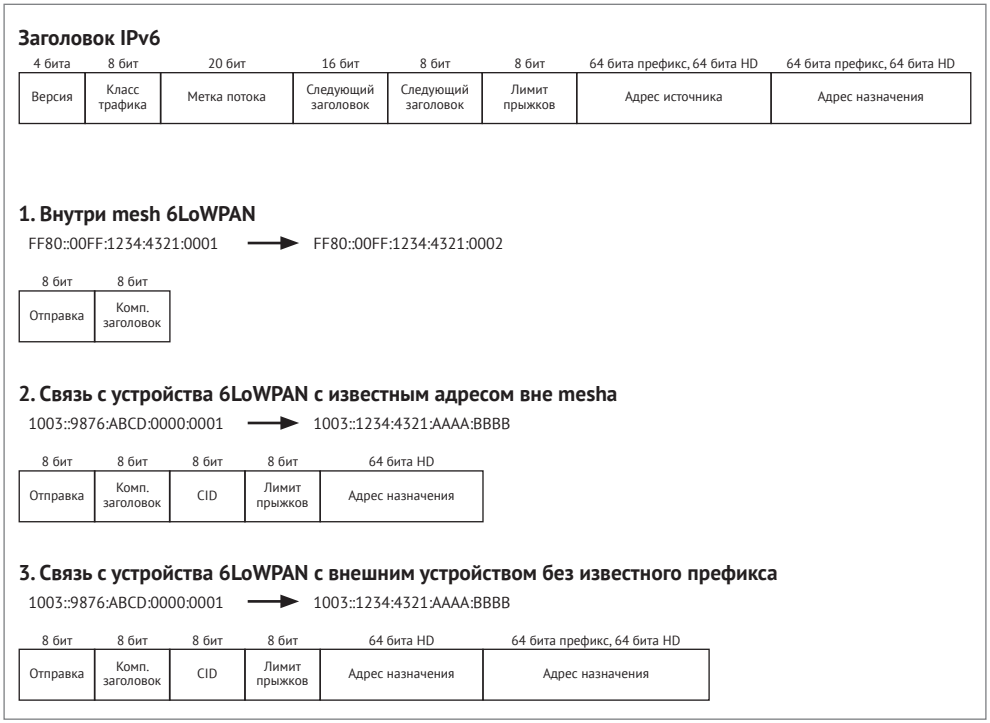


Рис. 6.5 ❖ Сжатие заголовка в 6LoWPAN

Заголовок фрагментации включает поле Datagram Size, которое определяет общий размер нефрагментированных данных. Поле Datagram Tag идентифицирует набор фрагментов, принадлежащих полезной нагрузке, а смещение датаграммы указывает, где фрагмент находится в последовательности полезной нагрузки. Обратите внимание, что смещение дейтаграммы не используется для первого отправленного фрагмента, так как смещение новой последовательности фрагментов должно начинаться с нуля (рис. 6.6).

Фрагментация – это ресурсоемкая задача, которая требует обработки и энергетических затрат, которые могут налагать особые условия на сенсорный узел, работающий на батарее. Целесообразно ограничить размеры данных (на прикладном уровне) и использовать сжатие заголовка для уменьшения потребления мощности и ресурсов в большой сети.

Рис. 6.6 ❖ Заголовок фрагментации 6LoWPAN

Обнаружение соседей

Обнаружение соседей (ND) определяется RFC4861 как протокол маршрутизации с одним переходом. Это формальный контракт между соседними узлами в сети, и он позволяет узлам общаться друг с другом. ND – это процесс обнаружения новых соседей, поскольку сеть может расти, сокращаться и трансформироваться, что приводит к новым и изменяющимся соседним отношениям. В ND есть два основных процесса и четыре основных типа сообщений:

- **поиск соседей** включает в себя фазы регистрации соседей (NR) и подтверждения соседей (NC);
- **поиск маршрутизаторов** включает в себя факультативные запросы к маршрутизатору (RS) и рекламирование маршрутизатора (RA).

Во время ND могут возникать конфликты. Например, если хост-узел отключается от маршрутизатора и устанавливает связь с другим маршрутизатором в той же сети. ND требуется для поиска дублирующихся адресов и недостижимых соседей в рамках спецификации. DHCPv6 может использоваться совместно для обнаружения соседа.

После того, как устройство, поддерживающее 802.15.4, загрузилось через физический и канальный уровни, 6LoWPAN может выполнить обнаружение соседей и выстроить сеть. Этот процесс будет действовать таким образом, как показано на рис. 6.7:

- поиск подходящей линии связи и подсети для беспроводной сети с малой мощностью;
- минимизация управляющего трафика, инициированного узлом;
- хост отправляет сообщение RS, чтобы запросить префикс сети;
- маршрутизатор отвечает префиксом;
- хост назначает локальный адрес одноадресной связи (FE80 :: IID);
- хост, передающий этот локальный адрес одноадресной связи в сообщении NR в mesh-сеть;
- выполнение **Duplicate Address Detection (DAD)**, ожидая NC в течение установленного времени. Если истечет время ожидания, предполагается, что адрес не используется.

После того, как хост настроен, он может начать общаться через интернет с уникальным адресом IPv6.

Если используется сетчатая маршрутизация, локальный адрес связи, полученный на шаге 5, может использоваться для связи с любым другим узлом

в сети 6LoWPAN. В схеме маршрутизации локальный адрес связи может использоваться для связи только с узлами на один прыжок. Для чего-то большего, чем один прыжок, требуется полный адрес маршрутизации.

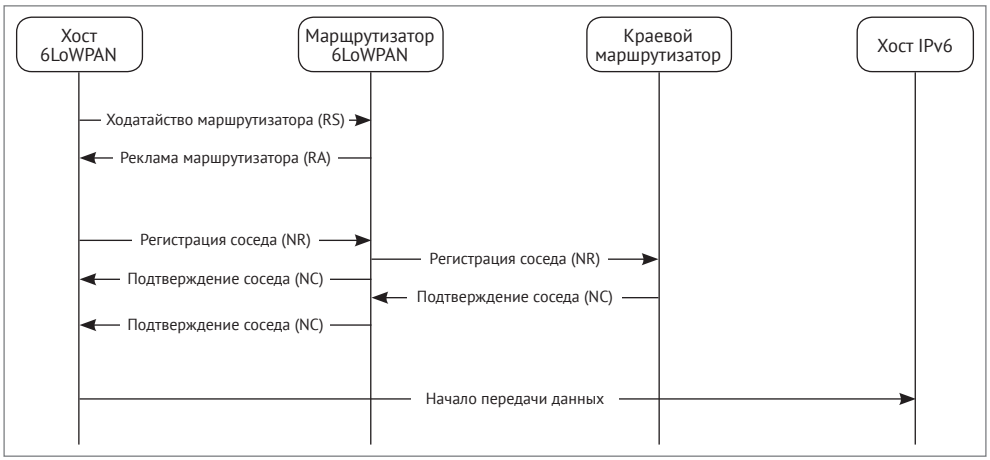


Рис. 6.7 ❖ Упрощенная последовательность обнаружения соседей из узла сетки 6LoWPAN через маршрутизатор mesh-сети на пограничный маршрутизатор и затем в глобальную сеть.

Безопасность 6LoWPAN

Поскольку в системе WPAN легко подслушивать и слушать сообщения, 6LoWPAN обеспечивает безопасность на нескольких уровнях. На уровне 802.15.4 два протокола, 6LoWPAN использует шифрование данных AES-128. Кроме того, 802.15.4 предоставляет счетчик с режимом CBC-MAC (CCM) для обеспечения шифрования и проверки целостности. Большинство наборов микросхем, которые обеспечивают сетевой блок 802.15.4, также включает механизм аппаратного шифрования для повышения производительности.

На третьем уровне (сетевой уровень) протокола 6LoWPAN имеет возможность использовать стандартную безопасность IPsec (RFC4301). Это включает в себя:

- **обработчик проверки подлинности (AH)**: как определено в RFC4302 для защиты целостности и аутентификации;
- **инкапсулирование полезной нагрузки безопасности (ESP)**: в RFC4303 добавляется шифрование для обеспечения конфиденциальности в пакетах.

ESP на сегодняшний день является наиболее распространенным форматом защищенного пакета уровня 3. Кроме того, режим ESP определяет повторное использование AES/CCM, используемого в аппаратных средствах уровня 2 для шифрования третьего уровня (RFC4309). Это делает безопасность уровня 3 подходящей для ограниченных узлов 6LoWPAN.

В дополнение к безопасности канального уровня, 6LoWPAN также использует протокол **Transport Layer Security (TLS)** для трафика TCP и **Datagram Transport Layer Security (DTLS)** для трафика UDP.

WPAN с IP – Thread

Thread – относительно новый сетевой протокол для IoT и основан на IPV6 (6LoWPAN). Его основной целью является домашняя связь и домашняя автоматизация. Тема была запущена в июле 2014 г. с формированием Thread Group Alliance, в которую входят такие компании, как Alphabet (холдинговая компания Google), Qualcomm, Samsung, ARM, Silicon Labs, Yale (замки) и Tyco.

Основываясь на протоколе IEEE 802.15.4 и 6LoWPAN, он имеет общность с Zigbee и другими вариантами 802.15.4, но с существенной разницей в том, что Thread является IP-адресуемым. Этот IP-протокол основывается на прикладном и физическом уровне, предоставляемых 802.15.4, и таких функциях, как безопасность и маршрутизация из 6LoWPAN. Thread также основан на mesh-сети, что делает его привлекательным для домашних систем освещения с 250 устройствами в одной сети. Философия Thread заключается в том, что, разрешая IP-адресацию в самом маленьком из датчиков и систем домашней автоматизации, можно уменьшить энергопотребление, потому что не нужно сохранять состояние приложения, поскольку протокол использует дейтаграммы на сетевом уровне. Это также подразумевает, что граничный маршрутизатор, на котором размещается сеть потоков Thread, не должен обрабатывать протоколы прикладного уровня и может снизить свою потребность в мощности и обработке. Наконец, будучи совместимым с IPV6, он по своей сути безопасен при шифровании всех сообщений с использованием стандарта расширенного шифрования (AES). В цепочке потоков может существовать до 250 узлов с полностью зашифрованным транспортом и аутентификацией. Обновление программного обеспечения позволяет уже существующему устройству 802.15.4 быть совместимым с Thread.

Архитектура и топология Thread

Основываясь на стандарте IEEE 802.15.4-2006, Thread использует спецификацию для определения уровней **управления доступом (MAC)** и физического (PHY). Он работает со скоростью 250 Кбит/с в полосе 2,4 ГГц.

С точки зрения топологии, Thread устанавливает связь с другими устройствами через граничный маршрутизатор (обычно это сигнал Wi-Fi в домашнем хозяйстве). Остальная часть сообщения основана на 802.15.4 и образует самовосстанавливающуюся сетку. Пример такой топологии показан на рис. 6.8.

Ниже перечислены роли различных устройств в архитектуре Thread.

- **граничный маршрутизатор** по существу, является шлюзом. В домашней сети это будет кроссовер связи от Wi-Fi до Thread, он формирует точку входа в интернет из сети Thread, проходящей под граничным

маршрутизатором. Несколько граничных маршрутизаторов допустимы в соответствии со спецификацией Thread;

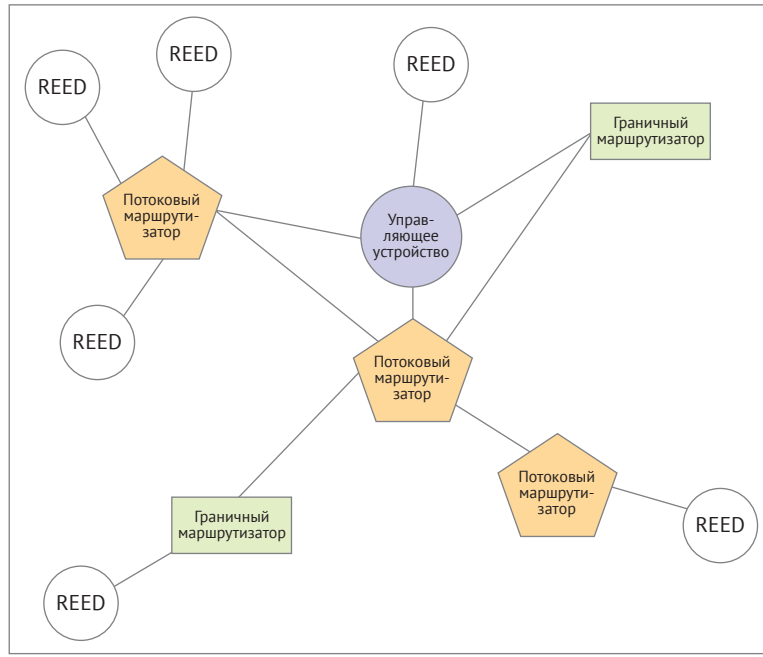


Рис. 6.8 ❖ Пример сетевой топологии Thread, содержащей граничные маршрутизаторы, маршрутизаторы Thread и подходящие устройства IoT, которые могут объединяться в mesh-сети. Взаимосвязи являются переменными и самовосстанавливающимися

- **ведущее устройство** управляет реестром назначенных идентификаторов маршрутизатора. Ведущий также контролирует запросы на **оконечные устройства, подходящие для маршрутизаторов (REED)**, для рекламирования на маршрутизаторы. Лидер может также выступать в роли маршрутизатора и иметь потомков – оконечные устройства. Протокол назначения адресов маршрутизатора – это **протокол ограничений приложений (CoAP)**. Информация о состоянии, которой управляет ведущее устройство, также может храниться на других маршрутизаторах потоков. Это позволяет обеспечить самовосстановление и переход на другой ресурс в случае, если лидер теряет связь;
- **маршрутизаторы потоков** управляют службами маршрутизации сети. Маршрутизаторы потоков никогда не входят в состояние ожидания, но спецификацией разрешается, чтобы они понижали свой уровень до REED;
- **REED**: хост-устройство, которое является REED, может стать маршрутизатором или лидером. REED не отвечают за маршрутизацию в mesh-

сети, если только они не назначаются маршрутизатором или лидером. REED также не могут передавать сообщения или присоединяться к сети. REED по существу являются оконечными точками или листьевыми узлами в сети;

- **оконечные устройства**: некоторые оконечные точки не могут стать маршрутизаторами. Эти типы REED имеют две другие категории, на которые они могут подписаться: **полноразмерные устройства (FED)** и **минимальные оконечные устройства (MED)**;
- **спящие оконечные устройства**: хост-устройства, которые вошли в состояние ожидания, обмениваются данными только с их связанным потоковым маршрутизатором и не могут передавать сообщения.

Стек протокола Thread

Thread будет использовать все преимущества 6LoWPAN и пользоваться преимуществами сжатия заголовков, адресации IPv6 и безопасности. Thread также использует схему фрагментации 6LoWPAN, как описано в предыдущем разделе, но добавляет два дополнительных компонента стека (табл. 6.2):

- дистанционно-векторная маршрутизация;
- создание mesh-связи.

Таблица 6.2. Стек протокола Thread

Стек протокола Thread	Упрощенная модель OSI
HTTP, CoAP, MQTT и др.	5. Прикладной уровень
Создание mesh-связи (MLE) и TLS/DTLS	4. Транспортный уровень
UDP	
Дистанционно-векторная маршрутизация	3. Сетевой уровень
IPV6	
6LoWPAN	
Уровень MAC IEEE 802.15.4	2. Канальный уровень
PHY IEEE 802.15.4	1. Физический уровень

Маршрутизация Thread

Thread использует обходную маршрутизацию, как описано в предыдущем разделе, в маршрутизации 6LoWPAN. В сети Thread разрешено до 32 активных маршрутизаторов. Обход маршрута основан на маршрутизации следующего перехода. Таблица основных маршрутов поддерживается стеком. Все маршрутизаторы имеют обновленную копию маршрутизации для сети.

Создание mesh-связи (MLE) – это способ обновления стоимости прохождения пути от одного маршрутизатора к другому в сети. Кроме того, MLE обеспечивает способ идентификации и настройки соседних узлов в сети и обеспечения их безопасности. Так как mesh-сеть может динамически расширяться, сжиматься и изменять форму, MLE обеспечивает механизм восстановления топологии. MLE разделит информацию о стоимости пути со всеми другими

маршрутизаторами в сжатом формате. Сообщения MLE будут транслироваться в сеть широковещательным образом по **протоколу многоадресной рассылки для сетей с низким энергопотреблением и потерями** (**MPL**).

Типичные сети 802.15.4 используют нахождение маршрута по требованию. Это может быть дорогостоящим (пропускная способность из-за наполнения сети сообщениями о поиске маршрута), а Thread пытается избежать этой схемы. Периодически сетевой маршрутизатор Thread будет обмениваться рекламными пакетами MLE со сведениями о стоимости связи со своими соседями, по сути, заставляя все маршрутизаторы иметь текущий список путей. Если маршрут прерывается (хост оставляет сеть Thread), маршрутизаторы пытаются найти следующий лучший путь к месту назначения.

Thread также измеряет качество связи. Помните, что 802.15.4 является WPAN и уровень сигнала может динамически меняться. Качество измеряется стоимостью соединения входящих сообщений для соседа со значением ноль (неизвестная стоимость) до значения три (хорошее качество). Табл. 6.3 суммирует отношение качества к стоимости. Эти качество и стоимость постоянно контролируются и, как уже упоминалось, периодически распространяются по сети для самовосстановления.

Таблица 6.3. Отношение качества к стоимости WPAN

Качество соединения	Стоимость соединения
0	Неизвестна
1	6
2	2
3	1

Адресация Thread

Чтобы найти маршрут к дочернему узлу, для пути просто проверяются старшие биты адреса дочернего элемента, чтобы найти адрес родительского маршрутизатора. На этом этапе источник передачи знает, как добраться до потомка, а также информацию о следующем переходе, чтобы начать маршрут.

Дистанционно-векторная маршрутизация используется для поиска путей к маршрутизаторам в сети Thread. Верхние 6 бит 16-разрядного адреса указывают маршрутизатор назначения – это префикс. Если нижние 10 бит адресата установлены в 0, конечным пунктом назначения является этот маршрутизатор. Иначе маршрутизатор назначения будет перенаправлять пакет на основе младших 10 бит (рис. 6.9).

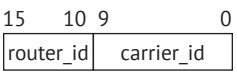


Рис. 6.9 ❖ 2-байтовый короткий адрес Thread по спецификации 802.15.4-2006

Если маршрут выходит за пределы сети Thread, граничный маршрутизатор будет сигнализировать лидерам конкретных префиксных данных, включив данные префикса, контекст 6LoWPAN, граничные маршрутизаторы и сервер DHCPv6. Эта информация передается с помощью пакетов MLE через сеть Thread.

В пределах сети Thread вся адресация основана на UDP. Если требуется повторная попытка, сеть Thread будет опираться на:

- **повторные попытки уровня MAC**: где каждое устройство, использующее подтверждения MAC, не получило ACK от следующего перехода;
- **повторные попытки уровня приложений**: где прикладной уровень будет обеспечивать свой собственный механизм повтора.

Обнаружение соседа

Протокол обнаружения соседей (ND) в потоке решает, к какой сети 802.15.4 присоединиться. Процесс выглядит следующим образом:

1) присоединение маршрутизатора контактов к входу;
2) соединяемое устройство сканирует все каналы и выдает запрос маяка на каждом канале. Ожидает ответа маяка;
3) если появится маяк, содержащий полезную нагрузку с **сетевым идентификатором набора услуг (SSID)** и сообщением о разрешении, устройство теперь присоединится к сети Thread;
4) как только устройство будет обнаружено, сообщения MLE будут переданы для идентификации соседнего маршрутизатора на устройстве. Этот маршрутизатор выполнит ввод в эксплуатацию. Существует два режима ввода в эксплуатацию:
   - **конфигурирование**: использует внеполосный метод ввода устройства в эксплуатацию. Позволяет подключать устройство к сети Thread, как только оно будет введено в сеть;
   - **создание**: создает сеанс ввода в эксплуатацию между устройством и комиссионным приложением, работающим на смартфоне, планшете или в интернете;
5) устройство, которое соединяется с родительским маршрутизатором и подключается к сети через MLE-обмен.

Устройство будет существовать как REED или оконечное устройство, а родительскому адресу назначается 16-битный короткий адрес.

Протоколы IEEE 802.11 и WLAN

Одной из первых, кто использовал ISM-диапазон, который FCC освободил для нелицензионного использования, была технология IEEE 802.11. IEEE 802.11 представляет собой набор протоколов с богатой историей и различными вариантами использования. 802.11 – это спецификация, определяющая **контроллер доступа к среде (MAC)** и физический уровень (PHY) сетевого стека. Определение и технические характеристики регулируются Комитетом стандартов

IEEE LAN/MAN. Wi-Fi – это определение WLAN на основе стандартов IEEE 802.11, но поддерживаемых и управляемых некоммерческим альянсом Wi-Fi.

802.11 обязана своим созданием NCR в 1991 г., который впервые разработал беспроводной протокол в качестве средства для сетевых кассовых аппаратов. Только в 1999 г., когда был создан Альянс Wi-Fi, эта технология стала повсеместной и широко распространенной на растущем рынке ПК и ноутбуков. Исходный протокол значительно отличается от современных протоколов 802.11 b/g/n/ac. Он поддерживает только скорость передачи данных 2 Мбит/с с прямой коррекцией ошибок.

Успех IEEE 802.11 можно объяснить многоуровневым подходом модели OSI. Простая замена уровней MAC и PHY уровнями IEEE 802.11 позволила бесшовно использовать существующую инфраструктуру TCP/IP. Сегодня почти на каждом мобильном устройстве, ноутбуке, планшете, встроенной системе, игрушке и видеоигре есть радиостанция IEEE 802.11. Тем не менее, 802.11 имеет легендарное прошлое, особенно в модели безопасности. Первоначальная модель безопасности 802.11 была основана на механизме безопасности UC Berkeley Wired Equivalent Privacy, которая позже оказалась ненадежной и была легко скомпрометирована. Несколько заметных лазеек, включая нарушение данных TJ Maxx через 802.11 WEP в 2007 г., привели к 45 миллионам украденных кредитных карт. Сегодня Wi-Fi Protected Access (WPA) и WPA2 с использованием предварительно разделенных ключей AES 256 бит, безусловно, ужесточили безопасность, и WEP редко используется.

В этом разделе будут подробно описаны некоторые различия в протоколах 802.11 и конкретная информация, относящаяся к архитектору IoT. Мы подробно рассмотрим текущий проект IEEE 802.11ac, а затем рассмотрим 802.11ah HaLow и 802.11p V2V, так как все три затрагивают интернет вещей.

Обзор и сравнение протоколов IEEE 802.11

Комитет стандартов IEEE LAN/MAN поддерживает и регулирует спецификации IEEE 802. Первоначальная цель 802.11 заключалась в предоставлении протокола канального уровня для беспроводной сети. В 2013 г. произошла эволюция с базового стандарта 802.11 до 802.11ac. С тех пор рабочая группа сосредоточилась на других областях, как показано в табл. 6.4. Специфические варианты 802.11 были рассмотрены для использования и сегментов, таких как соединение IoT с низкой мощностью / низкой пропускной способностью (802.11ah), связь между автомобилями (802.11p), повторное использование телевизионного аналогового RF-пространства (802.11af), экстремальная ширина полосы около метра для аудио/видео (802.11ad) и, конечно же, развитие стандарта 802.11ac (802.11ax).

Новые варианты предназначены для различных областей радиочастотного спектра или для уменьшения латентности и повышения безопасности при возникновении аварийных ситуаций на автомобилях. Таблица 6.4 отражает компромиссы между диапазоном, частотой и мощностью. В этом разделе мы рассмотрим аспекты, такие как модуляция, потоки MIMO и использование частот.

Таблица 6.4. Различные стандарты и спецификации IEEE 802.11 от основополагающих оригинальных спецификаций 802.11 до еще не утвержденного стандарта 802.11ax.

Протокол IEEE 802.11	Примечание	Дата выпуска	Частота (ГГц)	Полоса пропускания (МГц)	Скорость передачи данных на канал мин-макс (Мбит/с)	Позволенные потоки MIMO	Модуляция	Дистанция в помещении (м)	Дистанция снаружи (м)	Типичная рассеянная мощность на чип (мВт)
802.11	Первая разработка 802.11	01.1997	2,4	22	1–2	1	DSSS, FHSS	20	20	50
a	Выпущен одновременно с 802.11b, менее подвержен интерференции, чем 802.11b	09.1999	5	20	6–54	1	OFDM	30	120	50
			3,7				(SISO)		5000	
b	Выпущен одновременно с 802.11a, существенно повышена скорость по сравнению с 802.11a и улучшен диапазон	09.1999	2,4	22	1–11	1	DSSS (SISO)	50	150	7–50
g	Увеличена скорость по сравнению с 802.11b	06.2003	2,4	20	6–54	1	OFDM. DSSS (SISO)	38	140	50
n	Поддержка нескольких антенн для улучшения скорости и расширения диапазона	10.2009	2,4/5	20	7,2–72,2	4	OFDM (MIMO)	70	250	40
				40	15–150					
ac	Улучшены производительность и покрытие по сравнению с 802.11n. Уже каналы и улучшенная модуляция. Допускается работа с несколькими пользователями посредством MU-MIMO. Добавлена технология формирования луча	12.2013	5	20	7,2–96,3	8	OFDM (MU_MIMO)	35	35	40
				40	15–200					
				80	32,5–433,3					
				160	65–866,7					
ah	"WiFi HaLow" Предназначен для сетей и датчиков Интернета вещей. Очень низкая мощность и более широкий диапазон	12.2016	2,4/5	1–16	347	4	OFDM	1000	1000	tbd but goal is low power

p	«Беспроводной доступ в транспортной среде» «Интеллектуальные транспортные системы». Выделенная короткодиапазонная связь: службы передачи, предупреждение коллизий, транспортные сети	06.2009	5,9	10	27	1	OFDM	Н/д	400–1000	40
af	«Белый WiFi» или «Super WiFi» Разворачивается на неиспользуемом диапазоне телевизионных частот для обеспечения связью «Последняя миля» устройств в Индии, Кении, Сингапуре, США и Великобритании	11.2013	0.470–0,710	6–8	568	4	OFDM	Н/д	6000–100 000	tbd
ad	Альянс WiGig. Беспроводная связь 60 ГГц для видеоустройств и проекторов. Передача высококачественного аудио и видео трафика без кабелей	12.2012	60	2160	4260	>10	SC,OFDM (MU_MIMO)	10	10	tbd
ax	«Высокоэффективная беспроводная сеть (HEW)» Следующее поколение сетей 802.11. 4-кратное увеличение емкостей по сравнению с 802.11ac. Средний прирост скорости в 4 раза на пользователя по сравнению с 802.11ac. Обратная совместимость с сетями 802.11a/b/g/n/ac. Сценарии плотного развертывания	2019	2,4/5	20 40 80 160	450–10 000	8	OFDMA (MU_MIMO)	35	35	tbd

Архитектура IEEE 802.11

Протокол 802.11 представляет семейство беспроводной радиосвязи на основе различных методов модуляции в полосах ISM 2,4 ГГц и 5 ГГц нелицензионного спектра. 802.11b и 802.11g находятся в диапазоне 2,4 ГГц, а 802.11n и 802.11ac – в диапазоне 5 ГГц. В последней главе подробно описывался диапазон 2,4 ГГц и разные протоколы, которые находятся в этом пространстве. Wi-Fi восприимчив к тем же шумам и интерференции, что и Bluetooth и Zigbee, и использует различные методы для обеспечения надежности и отказоустойчивости.

С точки зрения стека протоколы 802.11 находятся на канальном уровне (один и два) модели OSI, как показано в табл. 6.5.

Таблица 6.5. Стек IEEE 802.11ac.

<table>
<tr><th colspan="8">Стек протокола 802.11</th><th>Упрощенная модель OSI</th></tr>
<tr><td colspan="8" rowspan="3">Уровень приложений</td><td>7. Прикладной уровень</td></tr>
<tr><td>6. Представительский уровень</td></tr>
<tr><td>5. Сеансовый уровень</td></tr>
<tr><td colspan="8">Транспортный уровень</td><td>4. Транспортный уровень</td></tr>
<tr><td colspan="8">Сетевой уровень</td><td>3. Сетевой уровень</td></tr>
<tr><td colspan="8">Логический контроль соединений</td><td rowspan="2">2. Канальный уровень</td></tr>
<tr><td colspan="8">Подуровень MAC</td></tr>
<tr><td>802.11
2,4 ГГц
FHSS
1 Мб/с,
2 Мб/с</td><td>802.11
2,4 ГГц
DHSS
1 Мб/с,
2 Мб/с</td><td>802.11
ИК
1 Мб/с,
2 Мб/с</td><td>802.11a
5 ГГц
OFDM
6, 9, 12,
18, 24,
36, 48,
54 Мб/с</td><td>802.11b
2,4 ГГц
DSSS
1, 2, 5,5,
11 Мб/с</td><td>802.11g
2,4 ГГц
OFDM
1, 2, 5,5,
11 и 6,
9, 12, 18,
24, 36, 48,
54 Мб/с</td><td>802.11n
2,4 ГГц
OFDM
От 1 до
450 Мб/с</td><td>802.11ac
MU-MIMO
5 ГГц
OFDM
200, 400,
433, 600,
866,
1300 Мб/с</td><td>1. Физический уровень</td></tr>
</table>

Стек включает различные PHY из более старых спецификаций 802.11, таких как оригинальные PHY 802.11 (включая инфракрасные), a, b, g и n. Это должно обеспечить обратную совместимость между сетями. Большинство чипсетов включают в себя всю коллекцию PHY, и трудно найти отдельно часть с более старым PHY.

Системы 802.11 поддерживают три основные топологии:

- **инфраструктура –** в этой форме **станция (STA)** относится к устройству конечной точки 802.11 (например, смартфону), которое осуществляет связь с центральной **точкой доступа (AP)**. AP может быть шлюзом к другим сетям (WAN); маршрутизатором или истинной точке доступа в более крупной сети. Это также известно как **служба базовой настройки инфраструктуры (BSS)**. Эта топология является топологией звезды;

- **по ситуации** – узлы 802.11 могут формировать так называемый **независимый базовый набор (IBSS)**, где каждая станция связывается и управляет интерфейсом с другими станциями. В этой конфигурации не используется точка доступа или топология звезды. Это одноранговая топология;
- **система распределения (DS)**: DS объединяет две или более независимых сетей BSS через межсетевые соединения точек доступа.

IEEE 802.11ah и IEEE 802.11s поддерживают формы топологии mesh-сети.

На рис. 6.10 приведены примеры трех основных топологий архитектуры IEEE 802.11.

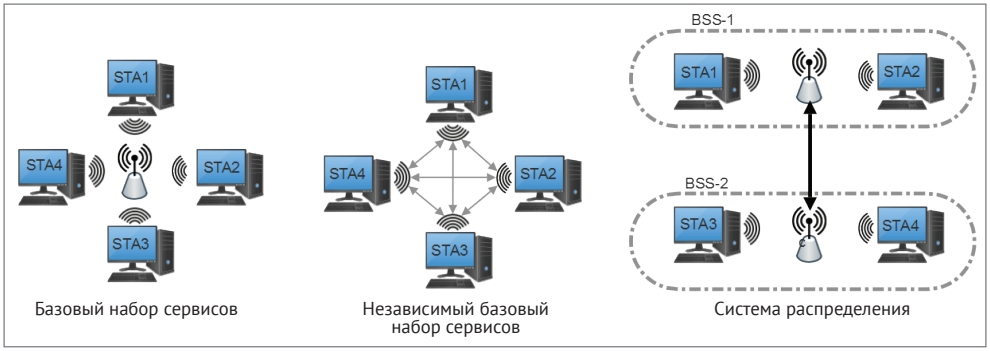


Рис. 6.10 ❖ Сетевые архитектуры 802.11. BSS, IBSS и система распределения объединяют две независимые BSS

В общей сложности протокол 802.11 позволяет использовать до 2007 STA, ассоциированных с одной точкой доступа. Это актуально, когда мы будем исследовать другие протоколы, такие как IEEE 802.11ah для IoT, далее в этой главе.

Распределение спектра IEEE 802.11

Первый протокол 802.11 использовал спектр в диапазонах ISM с частотой 2 ГГц и 5 ГГц и равномерно распределенные каналы, разнесенные примерно на 20 МГц друг от друга. Полоса пропускания канала составляла 20 МГц, но позже поправки от IEEE позволили работать и с полосой в 5 МГц и 10 МГц. В США 802.11b и g позволяют использовать одиннадцать каналов (другие страны могут поддерживать до четырнадцати). На рис. 6.11 показано разделение каналов. Три канала не перекрываются (1, 6, 11).

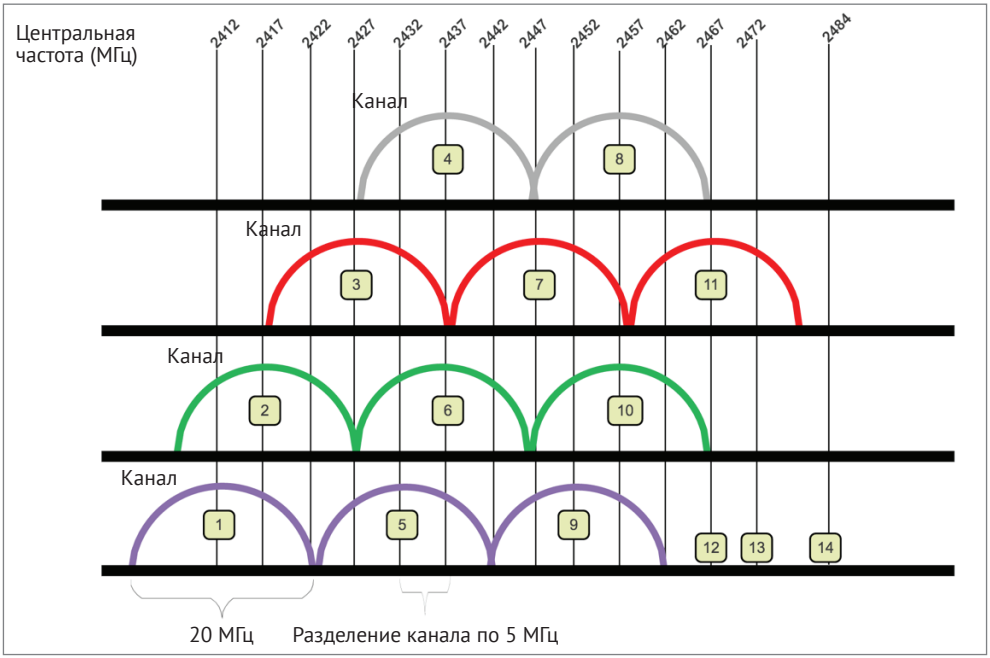


Рис. 6.11 ❖ Частотный диапазон 2,4 ГГц 802.11 и комбинация неперекрывающихся каналов. Обратите внимание на разделение каналов 5 МГц по 14 каналам с шириной 20 МГц

802.11 определяет спектральную маску, которая определяет разрешенное распределение мощности по каждому каналу. Спектральная маска требует, чтобы сигнал ослаблялся до определенных уровней (от его максимальной амплитуды) при заданных смещениях частоты. При этом сигналы будут стремиться излучать в соседние каналы. 802.11b с использованием **прямой последовательности распространения спектра (DSSS)** имеет совершенно иную спектральную маску, чем 802.11n, используя **мультиплексирование с ортогональным частотным разделением (OFDM)**. OFDM имеет намного более плотную спектральную эффективность и, следовательно, также поддерживает гораздо большую пропускную способность. Ниже приведены различия каналов и модуляции между 802.11 b, g и n. Ширина канала ограничивает количество одновременных каналов от 4 до 3 и до 1. Форма сигнала также изменяется между DSSS и OFDM. OFDM намного плотнее и, следовательно, способен обеспечить большую пропускную способность (рис. 6.12).

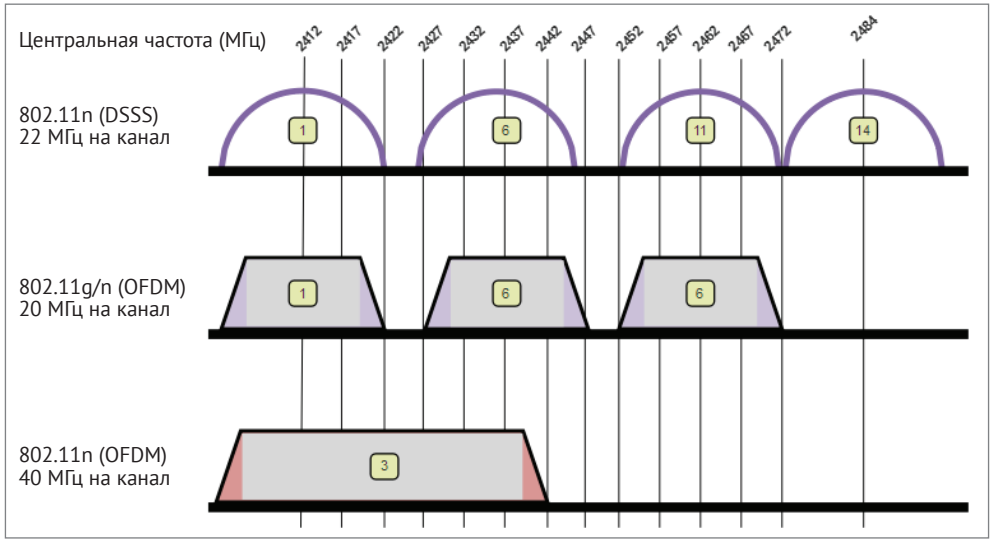


Рис. 6.12 ❖ Различия между 802.11b, g и n с использованием DSSS по сравнению с OFDM и шириной канала

Хотя в диапазоне 2,4 ГГц имеется 14 каналов, использование каналов регулируется регионом и страной. Например, Северная Америка позволяет использовать каналы с 1 по 11, Япония разрешает все 14 каналов для 802.11b и от 1 до 13 для 802.11g/n, Испания допускает только каналы 10 и 11, а Франция допускает от 10 до 13. Распределение варьируется и дизайнеры должны знать об ограничениях в стране. IEEE использует номенклатуру regdomain для описания ограничений по каналу, мощности и времени в стране, которые влияют на PHY.

Методы модуляции и кодирования IEEE 802.11

В этом разделе подробно описаны методы модуляции и кодирования в протоколе IEEE 802.11. Эти методы не уникальны для 802.11; они также применяются к протоколам 802.15 и, как мы увидим, к сотовым протоколам. Методы скачкообразной перестройки частоты, модуляции и фазового сдвига являются фундаментальными методами, которые должны понимать архитекторы как различные техники балансировки, интерференции и пропускной способности.

Цифровые данные, передаваемые радиочастотным сигналом, должны быть преобразованы в аналоговые. Это происходит в PHY независимо от того, какой радиочастотный сигнал описывается (Bluetooth, Zigbee, 802.11 и т. д.). Аналоговый несущий сигнал будет модулироваться дискретным цифровым сигналом. Это формирует так называемый символ или алфавит модуляции. Простым способом представить символьную модуляцию является фортепиано с четырьмя клавишами. Каждая клавиша представляет два бита (00, 01, 10, 11). Если вы можете нажимать 100 клавиш в секунду, это означает, что вы можете передавать 100 символов в секунду. Если каждый символ (тон от фортепиано) представля-

ет два бита, то он эквивалентен модуляции 200 бит/с. Хотя существует множество форм кодирования символов, три основные формы включают:

- **Amplitude Shift Keying (ASK)** – это форма амплитудной модуляции. Двоичный 0 представлен одной формой амплитуды модуляции и 1 – другой амплитудой. Простая форма показана на рис. 6.13, но более сложные формы могут представлять данные в группах с использованием дополнительных уровней амплитуды;
- **Frequency Shift Keying (FSK)** – этот метод модуляции модулирует несущую частоту, чтобы представить 0 или 1. Простейшей формой, показанной на рис. 6.13, является **бинарный сдвиг частоты (BPSK)**, который используется в протоколах 802.11 и других протоколах. В последней главе мы говорили о Bluetooth и Z-Wave: эти протоколы используют форму FSK, называемую **Gaussian Frequency Shift Keying (GFSK)**, которая пропускает данные через фильтр Гаусса, который сглаживает цифровой импульс (–1 или +1) и придает ему форму, чтобы ограничить спектральную ширину;
- **Phase Shift Keying (PSK)** – модулирует фазу опорного сигнала (сигнала несущей). Используется в основном в тегах 802.11b, Bluetooth и RFID. PSK использует конечное число символов, представленных в виде разных фазовых изменений. Каждая фаза кодирует равное количество бит. Картина битов образует символ. Получателю нужен контрастирующий опорный сигнал, и нужно вычислить разницу, чтобы извлечь символы и затем демодулировать данные. Альтернативный метод не требует опорного сигнала для приемника. Приемник проверит сигнал и определит, есть ли изменение фазы без ссылки на вторичный сигнал. Это называется **дифференциальной фазовой манипуляцией (DPSK)** и используется в 802.11b.

На рис. 6.13 изображены различные методы кодирования.

Следующей формой модуляции является иерархическая модуляция, в частности, **квадратурная амплитудная модуляция (QAM)**. Диаграмма созвездий на рис. 6.14 представляет собой кодирование в 2D (декартовой системе). Длина любого одного вектора представляет собой амплитуду, а угол к точке созвездия представляет собой фазу. Вообще говоря, существует больше фаз, которые могут быть закодированы, чем амплитуды, как показано на рис. 6.14 созвездий 16-QAM. 16-QAM имеет три уровня амплитуды и 12 общих фазовых углов. Это позволяет кодировать 16 бит. 802.11a и 802.11g могут использовать 16-QAM и даже более высокую плотность 64-QAM. Очевидно, что чем плотнее созвездие, тем больше может быть кодирование и тем выше пропускная способность.

Рисунок 6.14 иллюстрирует процесс кодирования QAM пиктографически. Слева находится представление диаграммы созвездий 16 точек (16-QAM). Существует 3 уровня амплитуды, обозначенных длиной векторов и 3 фазами на квадрант, обозначенными углом вектора. Это позволяет создавать 16 символов. Эти символы отражаются при изменении фазы и амплитуды генерируемого сигнала. Справа находится диаграмма сигналов для примера 8-QAM, показывающая изменяющиеся фазы и амплитуды, которые представляют 3-битный (8-значный) алфавит модуляции.

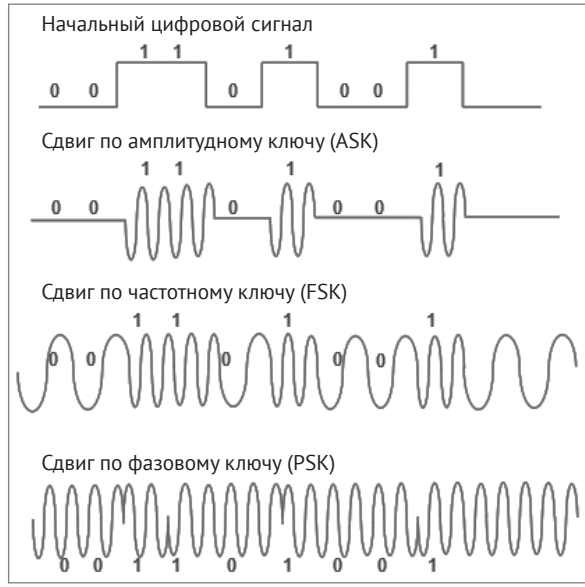


Рис. 6.13 ❖ Различные формы кодирования символов с использованием методов модуляции: амплитудная модуляция, частотная модуляция и фазовая модуляция. Обратите внимание, что фазовая модуляция меняет фазу для каждой встреченной «1»

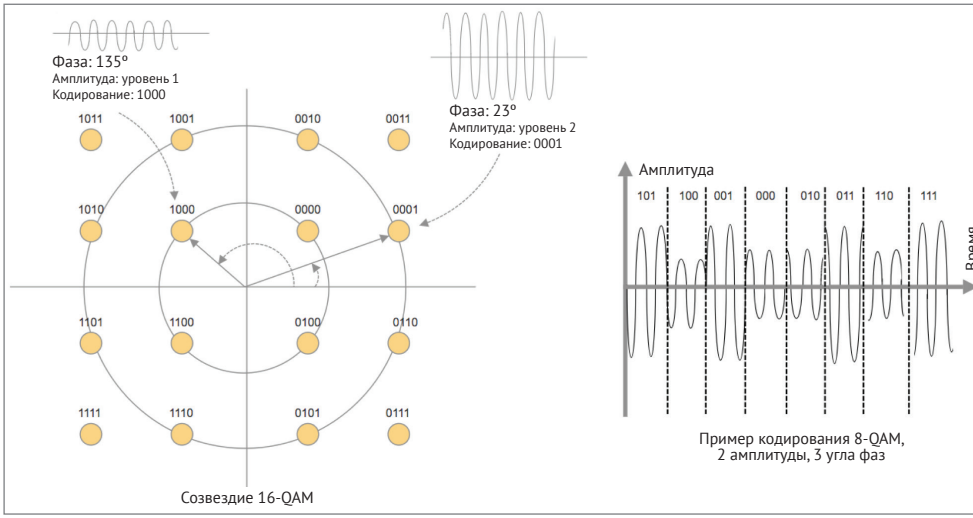


Рис. 6.14 ❖ Квадратурная амплитудная модуляция (QAM). Слева: Созвездие 16-QAM. Справа: кодировка формы волны 8-QAM

QAM имеет практические ограничения. Позже мы увидим очень плотные созвездия, которые радикально увеличивают пропускную способность, куда можно добавить только определенное количество фазовых углов и амплитуд. Шум, создаваемый **аналого-цифровыми преобразователями (АЦП)** и **цифроаналоговыми преобразователями (ЦАП)**, приведет к ошибкам квантования и шуму и потребует выборки сигналов на очень высоких скоростях. Кроме того, **отношение сигнал/шум (SNR)** должно превышать определенное значение для достижения хорошего **соотношения битовых ошибок (BER)**.

В стандартах 802.11 используются различные методы подавления помех, которые существенно распространяют сигнал по полосе пропускания:

- **скачкообразно изменяемый спектр частоты (FHSS)** – распространяет сигнал по 79 неперекрывающимся каналам с шириной 1 МГц в диапазоне ISM 2,4 ГГц. Использует генератор псевдослучайных чисел, чтобы начать процесс перескакиваний. Время задержки зависит от минимального времени, которое канал использует перед скачкообразной перестройкой (400 мс). Частотные перескоки были также описаны в последней главе и являются типичной схемой распространения сигналов;
- **спектр с прямой последовательностью –** сначала используется в протоколах 802.11b и имеет каналы шириной 22 МГц. Каждый бит представлен несколькими битами в передаваемом сигнале. Передаваемые данные умножаются генератором шума. Это будет эффективно распространять сигнал по всему спектру равномерно с использованием псевдослучайной последовательности чисел (так называемый *псевдошумовой PN*-код). Каждый бит передается с 11-битной *последовательностью измельчения* (фазовый сдвиг). Результирующий сигнал представляет собой XOR бит и 11-разрядную случайную последовательность. DSSS обеспечивает около 11 миллионов символов в секунду, когда мы рассматриваем скорость измельчения;
- **OFDM**: используется в IEEE 802.11a и более новых протоколах. Этот метод делит один канал 20 МГц на 52 подканала (48 для данных и четыре для синхронизации и мониторинга) для кодирования данных с использованием QAM и PSM. Для генерации каждого символа OFDM используется **быстрое преобразование Фурье (FFT)**. Набор резервных данных окружает каждый подканал. Этот избыточный диапазон данных называется **интервалом охраны (GI)** и используется для предотвращения **межсимвольной интерференции (ISI)** между соседними поднесущими. Обратите внимание, что поднесущие очень узкие и не имеют охранных полос для защиты сигнала. Это было сделано намеренно, поскольку каждая поднесущая одинаково разнесена с обратным временем символа. То есть все поднесущие переносят полное количество синусоидальных циклов, которые при демодуляции будут суммироваться до нуля. Из-за этого дизайн прост и не требует дополнительной стоимости полосовых фильтров. IEEE 802.11a использует 250 000 символов в секунду. OFDM, как правило, более эффективен и плотен (следовательно, имеет большую пропускную способность), чем DSSS, и используется в новых протоколах.

Меньше символов в секунду имеет преимущество в ситуациях, когда есть отражения сигналов на стенах и окнах. Так как отражения вызовут так называемое многопутевое искажение (копии символа попадают в приемник в разное время), более медленная скорость передачи символов позволяет больше времени передавать символ, и существует большая устойчивость к задержке распространения. Однако, если устройство движется, могут быть эффекты Доплера, которые влияют на OFDM больше, чем на DSSS. Другие протоколы, такие как Bluetooth, используют миллион символов в секунду.

На рис. 6.15 показана система OFDM с 52 поднесущими в двух каналах 20 МГц.

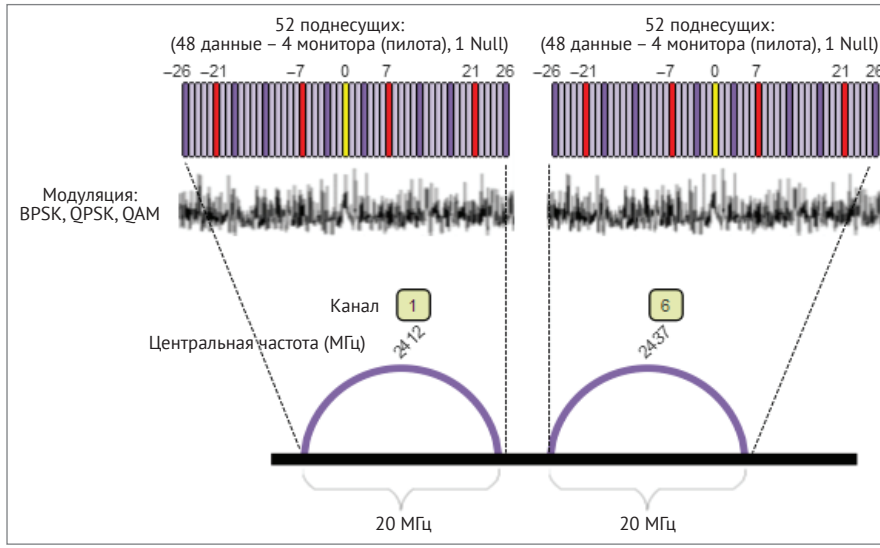


Рис. 6.15 ❖ Пример OFDM. Здесь канал подразделяется на 52 младших слота или поднесущих (каждый из которых переносит символ)

Набор различных типов модуляции, доступных для каждого стандарта, называется **схемой модуляции и кодирования (MCS)**. MCS представляет собой таблицу доступных типов модуляции, интервалов охраны и скорости кодирования. Используется ссылка на эту таблицу с индексом.

До выпуска 802.11b, в 802.11a использовались разные схемы кодирования. Каждая из них несовместима с другими. На рынке есть некоторая путаница в отношении разницы между ними, поскольку протоколы были выпущены почти одновременно.

IEEE 802.11 MIMO

MIMO – это аббревиатура, которая означает «множество вводов, множество выводов». MIMO использует ранее упомянутое явление RF, называемое многолучевым распространением. Многоточечная передача подразумевает, что сигналы будут отражаться от стен, дверей, окон и других препятствий. При-

емник будет видеть много сигналов, каждый из которых поступает в разное время по разным путям. Многоточечная передача имеет тенденцию искажать сигналы и вызывать помехи, что в конечном итоге ухудшает качество сигнала (этот эффект называется многолучевым замиранием). С добавлением нескольких антенн система MIMO может линейно увеличить пропускную способность данного канала, просто добавив больше антенн. Существуют две формы MIMO:

- **пространственное разнесение –** это относится к разнесению передачи и приема. Один поток данных передается на нескольких антеннах одновременно, используя пространственно-временное кодирование. Они обеспечивают улучшение отношения сигнал/шум и характеризуются улучшением надежности связи и охвата системы;
- **пространственное мультиплексирование**: используется для обеспечения дополнительной емкости данных путем использования нескольких путей для переноса дополнительного трафика, то есть – увеличение пропускной способности данных. По существу, один высокоскоростной поток данных будет разделен на несколько отдельных передач на разных антеннах.

В предыдущей главе мы представили методы скачкообразной перестройки частоты в сетях PAN, таких как Bluetooth. Частотный скачок является одним из способов преодоления проблемы затухания многолучевой передачи путем постоянного изменения углов многолучевого распространения. Это приводит к искажению размера радиочастотного сигнала. Системы Bluetooth обычно имеют одну антенну, что затрудняет работу MIMO. Что касается Wi-Fi, то только исходный стандарт 802.11 поддерживает форму скачкообразного изменения частоты (FHSS). Системы OFDM поддерживают блокировку канала и, следовательно, могут подвергаться проблеме многолучевого замирания.

Использование нескольких потоков влияет на общее потребление энергии. IEEE 802.11n включает в себя режим, позволяющий включать только MIMO, когда эффект будет иметь преимущество в производительности, что позволяет экономить электроэнергию в любое другое время. Альянс Wi-Fi требует, чтобы все продукты поддерживали по меньшей мере два пространственных потока для получения соответствия 802.11n.

WLAN будет разделять данные на несколько потоков, называемых пространственными потоками. Каждый переданный пространственный поток будет использовать другую антенну на передатчике. IEEE 802.11n позволяет использовать четыре антенны и четыре пространственных потока. Используя несколько потоков, отправляемых отдельно для антенн, расположенных на расстоянии друг от друга, пространственное разнесение в 802.11n дает некоторые подтверждения того, что по крайней мере один сигнал будет достаточно сильным, чтобы достичь приемника. Для поддержки функциональности MIMO необходимы как минимум две антенны. Потоковая передача также является независимой от модуляции. BPSK, QAM и другие формы модуляции работают с пространственной потоковой передачей. Цифровой процессор сигналов на передатчике и приемнике будет регулировать эффекты многолучевого рас-

пространения и задерживать передачу в прямой видимости всего лишь на достаточно продолжительное время, чтобы он идеально сочетался с линиями без прямой видимости. Это приведет к усилению сигналов.

Протокол IEEE 802.11n поддерживает реализацию однопользовательского MIMO (SU-MIMO) с четырьмя потоками, что означает, что передатчики будут работать в унисон, чтобы обмениваться данными с одним приемником. Здесь четыре передающие и четыре приемные антенны доставляют несколько потоков данных одному клиенту. Справа: эффект пространственного разнесения MIMO в 802.11n. Ниже приведен пример использования SU-MIMO и многолучевого распространения в 802.11n.

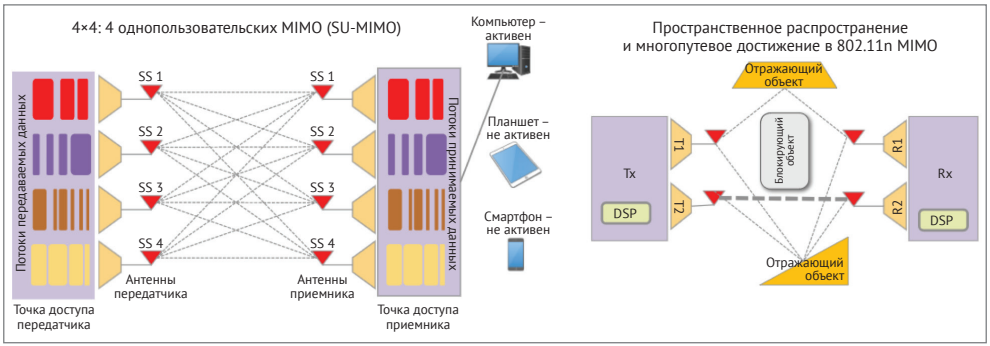


Рис. 6.16 ❖ Слева: иллюстрация SU-MIMO в IEEE 802.11n. Справа: эффект пространственного разнесения MIMO в 802.11n

На рис. 6.16 четыре передающие и четыре приемные антенны доставляют несколько потоков данных одному клиенту (SU-MIMO). Справа два передатчика, расположенные на некотором расстоянии друг от друга, сообщаются с двумя приемниками. Из-за отражений от двух передатчиков существуют несколько путей. Один путь линии прямой видимости сильнее, и он будет выбран более благоприятным. DSP на передатчиках и стороне приемника также уменьшают замирание многолучевой передачи путем объединения сигналов, поэтому результирующий сигнал имеет малое замирание.

Протокол IEEE 802.11 идентифицирует потоки MIMO по обозначению M x N : Z, где M – максимальное количество передающих антенн, а N – максимальное количество приемных антенн. Z – максимальное количество потоков данных, которые могут использоваться одновременно. Таким образом, MIMO 3 x 2 : 2 подразумевает наличие трех антенн передающего потока и двух антенн приемного потока, но может отправлять или принимать только два одновременных потока.

802.11n также представил дополнительную функцию формирования диаграммы направленности. 80211.n определяет два типа методов формирования луча: неявная обратная связь и явная обратная связь:

- **неявное формирование диаграммы направленности –** этот режим предполагает, что канал между формирователем луча (AP) и получателем луча (клиентом) является обратным (то же качество в обоих направлениях). Если это так, формирователь луча передает тренировочный кадр запроса и принимает пакет звонка. С помощью пакета звонка формирователь луча оценивает канал приемника и создает управляющую матрицу;
- **явное формирование диаграммы направленности –** в этом режиме получатель луча отвечает на тренировочный запрос, вычисляя свою собственную матрицу управления, и отправляет обратно матрицу в формирователь луча. Это более надежный метод.

На рис. 6.17 приведена диаграмма, иллюстрирующая эффекты формирования луча в ситуации без прямой видимости. В худшем случае сигналы поступают противоположными на 180° по фазе и аннулируют друг друга. При формировании луча сигналы могут корректироваться по фазе, чтобы усилить друг друга в приемнике.

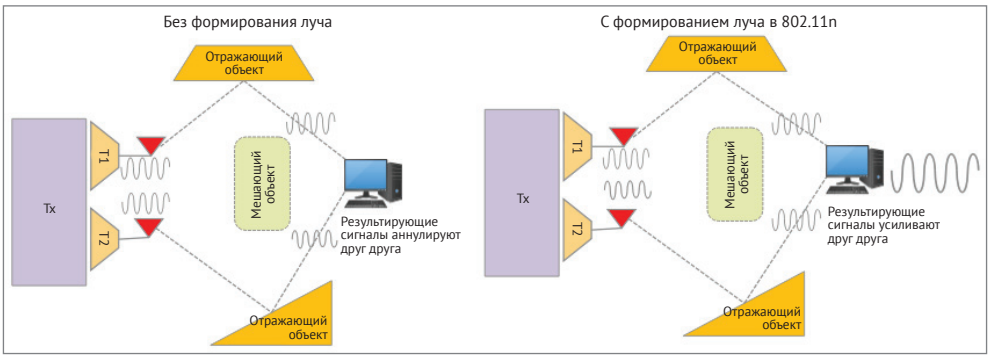


Рис. 6.17 ❖ Примеры системы с формированием луча и без него. В этом случае система не имеет прямой видимости и полагается на отражения для распространения сигнала

Формирование луча опирается на несколько разнесенных антенн для фокусировки сигнала в определенном месте. Сигналы могут быть отрегулированы по фазе и величине, чтобы достичь одного и того же местоположения и усилить друг друга, обеспечивая лучшую силу сигнала и дальность. К сожалению, 802.11n не стандартизовал один метод формирования луча и оставил это внедряющим. Различные производители использовали разные процессы и могли только гарантировать, что это будет работать с идентичным оборудованием. Таким образом, формирование луча в значительной степени не принималось во внимание во время разработки 802.11n.

Мы рассмотрим технологии MIMO во многих других областях, таких как 802.11ac, и в следующей главе о связи на большие расстояния с использованием сотовых 4G-LTE-радио.

Структура пакета IEEE 802.11

802.11 использует типичную структуру пакетов, которую мы видели ранее, с заголовками, данными полезной нагрузки, идентификаторами кадров и т. д. Начиная с организации кадра PHY, у нас есть три поля: преамбула, которая помогает на этапе синхронизации, заголовок PLCP, который описывает конфигурацию и характеристики пакета, такие как скорости обмена данными и данные MAC MPDC.

Каждая спецификация IEEE 802.11 имеет уникальную преамбулу и структурирована по количеству символов (описано ниже), а не по количеству бит для каждого поля. Примеры структур преамбулы следующие:

- **802.11a/g**: преамбула включает короткое поле обучения (два символа) и длинное поле обучения (два символа). Они используются поднесущими для синхронизации по времени и оценки частоты. Кроме того, преамбула включает в себя поле сигнала, которое описывает скорость передачи данных, длину и четность. Сигнал определяет, сколько данных передается в этом конкретном кадре.
- **802.11b**: преамбула будет использовать либо длинную последовательность из 144 бит, либо короткую последовательность из 72 бит. Заголовок будет включать в себя скорость обмена сигналами, режимы обслуживания, длину данных в микросекундах и CRC.
- **802.11n**: имеет два режима работы: Greenfield (HT) и смешанный (не HT). Greenfield можно использовать только там, где нет предшествующих систем. Режим без HT – режим совместимости с системами 802.11a/g и не обеспечивает лучшую производительность, чем a/g. Режим Greenfield позволяет осуществлять высокоскоростной транспорт.

На рис. 6.18 показана структура кадра пакета 802.11 PHY и канального уровня.

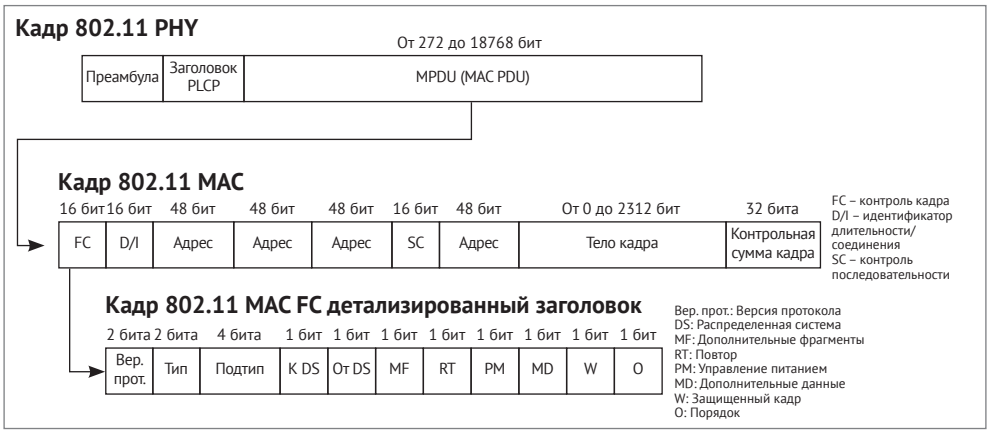


Рис. 6.18 ❖ Обобщенная структура кадра PHY и MAC 802.11

Структура кадра MAC показана на рис. 6.18. Кадр MAC содержит множество значащих полей. Подполе поля управления кадрами (поле FC) подробно описано ниже:

- **версия протокола** – указывает версию используемого протокола;
- **тип** – тип кадра WLAN как кадра контроля, данных или управления;
- **подтип** – дальнейшее определение типа кадра;
- **To DS и From DS**: кадры данных устанавливают один из этих битов в 1, чтобы указать, направляется ли кадр в систему распределения. Специальная сеть IBSS;
- **дополнительные фрагменты** – если пакет разделен на множество кадров, то каждый кадр, кроме последнего, будет иметь этот бит-набор;
- **повторить** – указывает, что кадр был повторно отправлен и помогает в разрешении передачи дубликатов кадров;
- **управление питанием**: указывает состояние питания отправителя. AP не могут устанавливать этот бит;
- **дополнительные данные**: AP будет использовать этот бит, чтобы помочь, когда STA находятся в режиме энергосбережения. Этот бит используется для буферизации кадров в системе распространения;
- **конфиденциальность, подобная проводному эквиваленту (WEP)**: установлено значение 1, когда фрейм расшифрован;
- **порядок**: если в сети используется строгий порядок, этот бит будет установлен. Фреймы могут быть отправлены не в порядке, а строгий режим принудительно задает передачу по порядку.

Перемещая кадр MAC из поля управления кадра, мы сначала исследуем бит ID продолжительности/соединения:

- **идентификатор продолжительности/соединения**: указывает продолжительность, свободный от конфликтов период и идентификатор ассоциации. Идентификатор ассоциации регистрируется во время первоначального установления связи Wi-Fi;
- **поля адреса**: 802.11 может управлять четырьмя MAC-адресами в следующем порядке:
  - **адрес 1**: получатель;
  - **адрес 2**: передатчик;
  - **адрес 3**: используется для фильтрации;
- **SC**: контроль последовательности – это 16-разрядное поле для порядка сообщений.

Протокол 802.11 имеет несколько типов фреймов, представленных полями типа и подтипа. Существует три основных типа: кадры управления, кадры контроля и кадры данных.

Кадры управления обеспечивают сетевое администрирование, безопасность и обслуживание. В табл. 6.6 определены типы кадров управления.

Таблица 6.6. Типы кадров управления пакета IEEE 802.11

Название кадра	Описание
Кадр аутентификации	STA отправит кадр проверки подлинности в AP, который отвечает своим собственным кадром проверки подлинности. Здесь общий ключ отправляется и проверяется с использованием ответа на вызов
Кадр запроса ассоциации	Передается из STA для запроса точки доступа для синхронизации. Содержит SSID, к которой STA хочет присоединиться, и другую информацию для синхронизации
Кадр ответа ассоциации	Передано от AP в STA, содержит сообщение о принятии или отклонении на запрос ассоциации. Если принято, идентификатор ассоциации будет отправлен в полезной нагрузке
Кадр маяка	Это периодический радиосигнал с AP. Включает SSID
Кадр деаутентификации	Передано из STA, желающей выйти из соединения с другой STA
Кадр дизассоциации	Передано из STA, желающей прекратить соединение
Кадр пробного запроса	Бродкаст от одного STA к другой STA.
Кадр ответа на пробу	Передано из AP в ответ на пробный запрос. Содержит информацию, такую как поддерживаемые скорости передачи данных
Кадр переассоциации	Используется, когда STA теряет мощность сигнала с одной точкой доступа, но находит другую AP, связанную с сетью, используя более сильный сигнал. Новая AP попытается связаться с STA и передать информацию, хранящуюся в буфере исходного AP
Кадр ответа на переассоциацию	Передано из AP с принятием или отклонением запроса на повторную передачу

Следующий основной тип кадра – это кадр управления. Контрольные кадры помогают обмениваться данными между STA (табл. 6.7).

Таблица 6.7. Контрольные кадры пакета IEEE 802.11

Название кадра	Описание
Кадр подтверждения (ACK)	Принимающая STA всегда будет получать данные ACK, если ошибок не было. Если отправитель не получает ACK по истечении установленного времени, отправитель будет повторно отправлять кадр
Запрос раскола (RTS)	Это часть механизма предотвращения столкновений. STA начнет с отправки сообщения RTS, если он хочет передать некоторые данные
Очистка для отправки кадра (CTS)	Ответ STA на кадр RTS. Запрос STA теперь может отправлять кадр данных. Это форма управления столкновениями. Значение времени используется для задержки передач других STA от STA, запрашивающей передачу

И, наконец, кадр данных. Это основная часть функции передачи данных протокола.

Работа IEEE 802.11

Как упоминалось ранее, STA считается устройством, оснащенным контроллером беспроводного сетевого интерфейса. STA всегда будет слушать активную связь в определенном канале. Первой фазой подключения к Wi-Fi является фаза сканирования. Существует два типа используемых механизмов сканирования:

- **пассивное сканирование –** эта форма сканирования использует маяки и пробные запросы. После выбора канала устройство, выполняющее сканирование, будет получать сигналы маяков и проб от ближайших STA. Точка доступа может передавать маяковый радиосигнал, и, если STA принимает передачу, она может продолжить присоединение к сети;
- **активное сканирование –** в этом режиме STA попытается найти точку доступа путем создания пробных запросов. Этот режим сканирования использует больше мощности, но позволяет быстрее присоединяться к сети. AP может ответить на пробный запрос с ответом на запрос пробы, который аналогичен сигналу маякового радиосигнала.

Точка доступа обычно будет транслировать маяк с фиксированным временным интервалом, называемым **временем передачи целевого маяка (TBTT)**. Обычно TBTT – это один раз каждые 100 мс.

Маяки всегда транслируются по самым низким базовым скоростям, чтобы гарантировать, что каждая STA в диапазоне имеет возможность принимать маяк, даже если он не может подключиться к этой конкретной сети. После обработки маяка следующей фазой подключения Wi-Fi является фаза синхронизации. Эта фаза необходима, чтобы клиенты были настроены на точку доступа. Пакет маяка содержит информацию, необходимую STA:

- **SSID –** идентификатор набора услуг. 1-32-символьное сетевое имя (это поле может быть скрыто, если установить длину SSID равным 0. Даже если оно скрыто, другие части кадра маяка передаются как обычно. Как правило, использование скрытой SSID не дает дополнительной сетевой безопасности;
- **BSSID –** базовый идентификатор набора услуг. Уникальные 48-битные следующие соглашения MAC-адреса уровня 2. Формируется комбинацией 24-битного уникального идентификатора организации и 24-разрядного идентификатора производителя радиочипа;
- **ширина канала –** 20 МГц, 40 МГц и т. д.;
- **страна** – список поддерживаемых каналов (для конкретной страны);
- **интервал маяка –** время TBTT, упомянутое ранее;
- **TIM/DTIM**: время пробуждения и интервалы для получения широковещательных сообщений – позволяет осуществлять расширенное управление питанием.
- **службы безопасности**: возможности WEP, WPA и WPA2.

Маяки представляют интересную концепцию, похожую на Bluetooth-маяки. Беспроводная связь Bluetooth обеспечивает гораздо большие возможности сообщений и гибкость в широковещательных передачах маяка, но есть ряд продуктов и услуг, которые также используют Wi-Fi-маяк.

Если STA обнаруживает AP или другую STA для установления соединения, она затем переходит к фазе аутентификации. Более подробно будет обсуж-

даться множество стандартов безопасности, используемых в 802.11, позже в этой главе.

Если процесс обеспечения безопасности и аутентификации завершается успешно, следующей фазой является ассоциация. Устройство отправит кадр запроса ассоциации на AP. Затем AP ответит кадром ответа ассоциации, который позволит STA присоединиться к сети или быть исключенной. Если STA включена, AP выдает идентификатор ассоциации клиенту и добавляет его в список подключенных клиентов.

На этом этапе данные могут обмениваться с AP и наоборот. По всем кадрам данных будет получено подтверждение.

Безопасность IEEE 802.11

В предыдущем разделе мы описали процесс ассоциации Wi-Fi-устройства для присоединения к сети. Одним из этапов была аутентификация. В этом разделе будут рассмотрены различные типы аутентификации, используемые в беспроводных сетях Wi-Fi, а также различные сильные и слабые стороны:

- **WEP –** конфиденциальность в проводном эквиваленте. Этот режим отправляет ключ в обычном тексте от клиента. Затем ключ шифруется и отправляется обратно клиенту. WEP использует ключи разного размера, но обычно они 128 бит или 256 бит. WEP использует общий ключ, что означает, что один и тот же ключ доступен для всех клиентов. Его можно легко скомпрометировать, просто прослушивая и просматривая все кадры аутентификации, возвращающиеся клиентам, входящим в сеть, для определения ключа, используемого для всех. Из-за слабости генерации ключа первые несколько байтов псевдослучайной строки могут выявить (вероятность 5%) часть ключа. Перехватывая от 5 до 10 миллионов пакетов, злоумышленник может с достаточной уверенностью получить достаточно информации для раскрытия ключа;
- **WPA –** защищенный доступ Wi-Fi (или WPA-Enterprise) был разработан как стандарт безопасности IEEE 802.11i для замены WEP и представляет собой программное / встроенное программное обеспечение, не требующее нового оборудования. Одно существенное различие заключается в том, что WPA использует **протокол целостности временного ключа (TKIP)**, который выполняет смешивание и повторный ключ для каждого пакета. Это означает, что каждый пакет будет использовать другой ключ для шифрования, в отличие от WEP. WPA начинается с создания ключа сеанса на основе MAC-адреса, временного ключа сеанса и вектора инициализации. Это достаточно загружает процессор, но выполняется только один раз за сеанс. Следующим шагом будет извлечение младших 16 бит принятого пакета с результатом бит, сгенерированным на первой фазе. Это 104-битный ключ для каждого пакета. Теперь данные могут быть зашифрованы;

- **WPA-PSK** – предварительный общий ключ WPA или WPA-Personal. Этот режим существует там, где нет инфраструктуры аутентификации 802.11. Здесь используется ключевая фраза как предварительный общий ключ. Каждая STA может иметь свой собственный предварительно открытый ключ, связанный с ее MAC-адресом. Это похоже на WEP, и недостатки уже обнаружены, если в предварительно разделяемом ключе используется слабая кодовая фраза;
- **WPA2** – это заменяет оригинальную разработку WPA. WPA2 использует AES для шифрования, что намного сильнее, чем TKIP в WPA. Это шифрование также называется режимом CTR с CBC-MAC Protocol или CCMP для краткости.

Для достижения высоких скоростей передачи в 802.11n необходимо использовать режим CCMP, или скорость передачи данных не превысит 54 Мбит/с. Кроме того, для использования логотипа торговой марки Wi-Fi Alliance требуется сертификация WPA2.

Протокол IEEE 802.11ac

IEEE 802.11ac – это беспроводная локальная сеть следующего поколения и продолжение семейства стандартов 802.11. IEEE 802.11ac был утвержден в качестве стандарта в декабре 2013 г. после пяти лет работы. Цель состоит в том, чтобы обеспечить многостанционную пропускную способность не менее 1 Гбит/с и пропускную способность одной линии 500 Мбит/с. Технология позволяет это за счет более широкой пропускной способности канала (160 МГц), большего количества пространственных потоков MIMO и модуляции с высокой плотностью (256-QAM). 802.11ac существует только в диапазоне 5 ГГц, но будет сосуществовать с предыдущими стандартами (IEEE 802.11a/n).

Специфика и отличия IEEE 802.11ac и IEEE 802.11n:

- минимальная ширина канала 80 МГц с максимальной шириной канала 160 МГц;
- восемь пространственных потоков MIMO:
  - внедрение MU-MIMO нисходящей линии связи с четырьмя нисходящими клиентами;
  - несколько STA с несколькими антеннами теперь могут передавать и принимать независимо друг от друга на нескольких потоках;
- 256-QAM факультативная модуляция с возможностью использования 1024-WAM;
- стандартизованная способность к формированию луча.

Многопользовательский MIMO заслуживает дополнительной информации. 802.11ac расширяет 802.11n от четырех пространственных потоков до восьми. Одним из основных факторов, способствующих скорости 802.11ac, является **мультиплексирование пространственного разделения (SDM)**, о котором упоминалось ранее. В сочетании с многопользовательскими или множественными клиентскими аспектами 802.11ac этот метод используется под названием **Spatial Diversity Multiple Access (SDMA)**. По существу, MU-MIMO в 802.11ac

является беспроводным аналогом сетевого коммутатора. На рис. 6.19 показана система 802.11ac 4×4: 4 MU-MIMO с 3 клиентами.

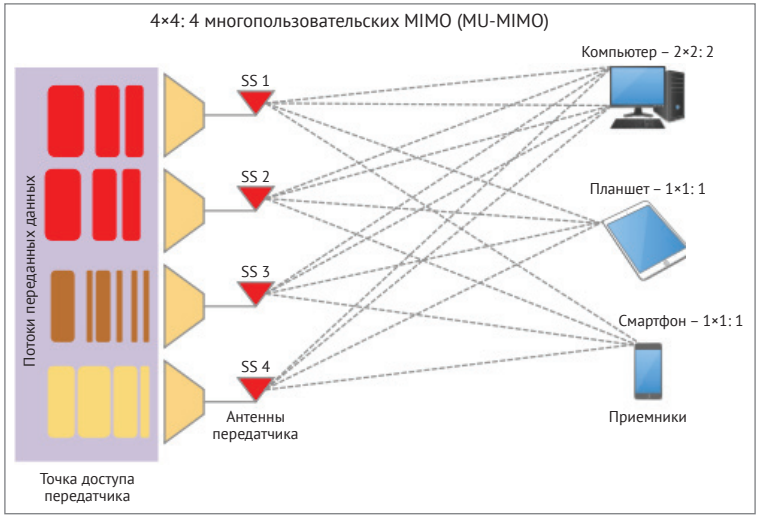


Рис. 6.19 ❖ Использование 802.11ac MU-MIMO

80211.ac также расширяет возможности модуляции от 64-QAM до 256-WAM. Это означает, что имеется 16 амплитудных уровней и 16 фазовых углов, требующих очень точного оборудования для реализации. 802.11n предоставляет шесть бит на символ, в то время как 802.11ac предоставляет полные восемь бит на символ.

Комитет по IEEE формально стандартизовал методы формирования диаграммы направленности. Например, комитет согласился с тем, что явная обратная связь является стандартным подходом к ассоциации формирования луча. Это позволит получать диаграммы направленности и преимущества производительности от нескольких поставщиков.

Увеличение полосы пропускания на канал (до 80 МГц с возможностью использования 160 МГц или двух блоков по 80 МГц) обеспечивает существенное увеличение пропускной способности в пространстве 5 ГГц. Теоретически, используя 8 x 8 : 8 устройство, ширину 160 МГц и 256-QAM-модуляцию, можно было бы обеспечить пропускную способность 6,933 ГБ/с в совокупности.

Транспорт-к-транспорту IEEE 802.11p

Транспортные сети (иногда называемые автомобильными сетями или VANET) являются спонтанными и неструктурированными, поскольку автомобиль перемещается по городу при взаимодействии с другими транспортными средствами и инфраструктурой. В этой модели сети используются модели передачи **автомобиль-автомобиль (V2V)** и **автомобиль-сеть (V2I)**.

В 2004 г. целевая группа 802.11p сформировала и разработала первый проект к апрелю 2010 г. 802.11p считается **выделенной короткой связью (DSRC)** в Министерстве транспорта США. Цель этой сети – обеспечить стандартную и безопасную систему V2V и V2I, используемую для обеспечения безопасности транспортных средств, сбора дорожных данных, статуса транспорта/предупреждений, помощи на дороге и электронной коммерции внутри транспортного средства.

Топология и общий пример для сети IEEE 802.11p показан на рис. 6.20. В сети есть два типа узлов. Сначала это **дорожное устройство (RSU)**, которое является устройством фиксированного местоположения, подобно точке доступа. Оно обслуживает мосты транспортных средств и перемещение устройств в интернет для использования служебных программ и доступа к доверенным органам. Другим типом узла является бортовой **блок (OBU)**, который находится в транспортном средстве. Он может взаимодействовать с другими OBU и фиксированными RSU, когда это необходимо.

OBU могут связываться с RSU и друг с другом для передачи данных о транспортных средствах и безопасности. RSU используются для подключения к службам приложений и доверенным авторитетам для аутентификации. Ниже приведен пример использования и топологии 802.11p.

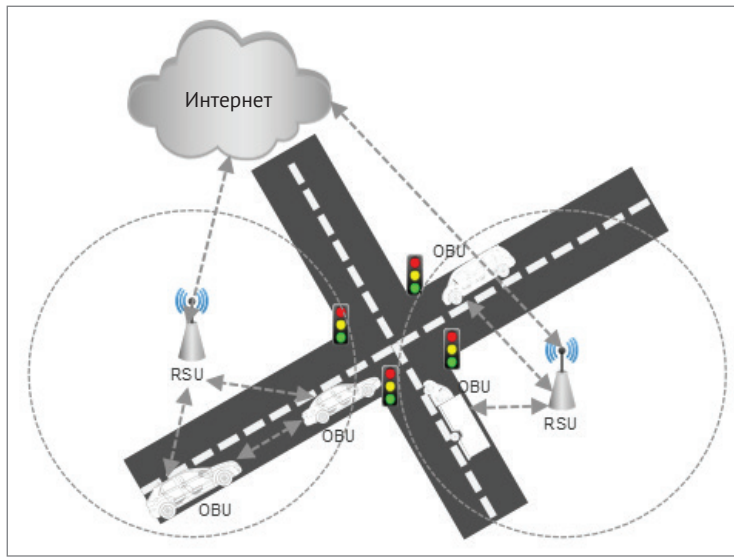


Рис. 6.20 ❖ Вариант использования IEEE 802.11p.
Показаны OBU в транспортных средствах и RSU с фиксированной инфраструктурой

Для транспортных систем в беспроводной связи существует несколько проблем. Есть повышенный уровень безопасности, необходимый для использования в автомобильной связи и управлении. В качестве некоторых вопросов

необходимо учитывать такие физические эффекты, как доплеровские сдвиги, эффекты задержки и надежные специальные сети.

Многое, что отличается от стандартов 802.11, заключается в обеспечении качества и дальности передачи по скорости передачи. Другими факторами являются изменения для уменьшения латентности для начала обмена данными. Ниже приведен краткий обзор особенностей IEEE 802.11p и отличия от стандартов IEEE 802.11a:

- ширина канала составляет 10 МГц вместо 20 МГц, используемых в 802.11a;
- IEEE 802.11p работает в полосе 75 МГц на частоте 5,9 ГГц. Это означает, что имеется в общей сложности семь доступных каналов (один канал управления, два критических и четыре канала услуг);
- он поддерживает половину битовых скоростей по сравнению с 802.11a; то есть 3/4,5/6/9/12/18/24/27 Мбит/с;
- у него такие же схемы модуляции, как BPSK/QPSK/16QAM/64QAM и 52 поднесущих;
- длительность символа стала двойной по сравнению с 802.11a: IEEE 802.11p поддерживает 8 мкс, а 11a поддерживает 4 мкс;
- охранный интервал составляет 1,6 мкс в 802.11p, а в 11a – 0,8 мкс;
- такие методы, как MIMO и формирование диаграммы направленности, не являются необходимыми или частью спецификации.

Каналы 75 МГц классифицируются так, как показано в табл. 6.8.

Таблица 6.8. Классификация каналов 75 МГц

Канал	172	174	176	178	180	182	184
Центральная частота (ГГц)	5,860	5,870	5,880	5,890	5,900	5,910	5,920
Цель	Критически важен для безопасности	Обслуживание	Обслуживание	Управление	Обслуживание	Обслуживание	Мощная общественная безопасность

Основная модель использования 802.11p – это быстрое создание и привязка к специальным сетям. Это соединение придет и уйдет, поскольку транспортные средства движутся на скоростях по шоссе в противоположных направлениях. В стандартной модели 802.11 BSS будет использовать топологию сети, которая требует синхронизации, ассоциации и аутентификации для создания беспроводной сети. 802.11p предоставляет BSS ID подстановочного знака в заголовке всех обмененных фреймов и позволяет начать обмен кадрами данных сразу после прибытия по каналу связи.

Протокол IEEE 802.11p вырос из 802.11a, но вносит важные изменения в отношении общей безопасности и безопасности транспортных средств. В табл. 6.9 показан стек протокола. Значительным отступлением от других стеков IEEE 802.11 является использование стандартов IEEE 1609.x для решения прикладных задач и для моделей безопасности. Полный стек называется беспроводным доступом в автомобильных средах (WAVE) и объединяет уровни PHY и MAC 802.11p с уровнями IEEE 1609.x.

Таблица 6.9. Стек протокола 802.11 в сравнении с моделью OSI

<table>
<tr><th colspan="2">Стек протокола 802.11</th><th>Упрощенная модель OSI</th></tr>
<tr><td colspan="2">IEEE 1609.1 (Safety and Traffic Efficiency-Applications)</td><td>7. Прикладной уровень</td></tr>
<tr><td colspan="2">IEEE 1609.2 WAVE Security Services</td><td rowspan="2">4. Транспортный уровень</td></tr>
<tr><td>TCP/UDP</td><td rowspan="2">IEEE 1609.3
WSMP</td></tr>
<tr><td>IPV6</td><td>3. Сетевой уровень</td></tr>
<tr><td colspan="2">Logical Link Control</td><td rowspan="2">2. Канальный уровень</td></tr>
<tr><td colspan="2">IEEE 1609.4 MAC SubLayer</td></tr>
<tr><td colspan="2">802.11p
5 ГГц
OF DM
3; 4,5; 6; 9; 12; 18; 24; 27 Мб/с</td><td>1. Физический уровень</td></tr>
</table>

Конкретные отличия, выделяемые в стеке, включают:

- **1609.1** – менеджер волновых ресурсов. Выделяет и обеспечивает ресурсы по мере необходимости;
- **1609.2** – определяет службы безопасности для сообщений приложений и управления. Этот уровень также обеспечивает два режима шифрования. Можно использовать алгоритм с открытым ключом, использующий подписи ECDSA. В качестве альтернативы доступен симметричный алгоритм, основанный на AES-128 в режиме CCM;
- **1609.3** – помогает установить соединение и управлять устройствами, совместимыми с WAVE;
- **1609.4** – обеспечивает многоканальную работу над уровнем MAC 802.11p.

Безопасность важна в VANET. Существуют потенциальные угрозы, которые могут напрямую влиять на общественную безопасность. Атаки могут возникать при трансляции фиктивной информации, влияющей на то, как другие транспортные средства реагируют или выполняют действия. Атака такого характера может быть произведена устройством-изгоем, передающим опасность на дороге, что приводит к остановке транспортных средств. Безопасность также должна учитывать частные данные о транспортных средствах, которые маскируются под другие транспортные средства и вызывают отказ в предоставлении услуг. Все это может привести к катастрофическим событиям и нуждаться в таких стандартах, как IEEE 1609.2.

Протокол IEEE 802.11ah

На основе архитектуры 802.11ac и PHY, 802.11ah – это вариант беспроводных протоколов, предназначенных для IoT. Проект пытается оптимизировать устройства с ограниченным сроком службы, которые требуют длительного действия батареи и могут оптимизировать диапазон и пропускную способность. 802.11ah также называется HaLow, что, по сути, является игрой слов с «ha», относящейся к «ah», подразумевая низкую мощность и более низкую частоту («low»). Составленные вместе, они образует производную от «hello».

Целью группы IEEE 802.11ah было создание протокола с расширенным диапазоном для связи в сельских районах и разгрузки мобильного трафика. Вторичной целью было использование протокола для беспроводной связи с низкой пропускной способностью в диапазоне субгигагерц. Спецификация была опубликована 31 декабря 2016 г. Архитектура больше всего отличается от других стандартов 802.11 следующим, в частности:

- работает в диапазоне 900 МГц. Это позволяет обеспечить хорошее распространение и проникновение через материалы и независимость от атмосферных условий;
- ширина канала изменяема и может быть установлена на 2, 4, 8 или 16 МГц;
- доступны разные методы модуляции, включая BPSK, QPSK, 16-QAM, 64-WAM и 256-QAM;
- модуляция на основе стандарта 802.11ac со специфическими изменениями. В общей сложности 56 поднесущих OFDM с 52, предназначенными для данных, и четыре, предназначенные для пилотных тонов;
- общая длительность символа составляет 36 или 40 мкс;
- поддерживает формирование луча SU-MIMO;
- быстрая связь для сетей тысяч STA, использующих два разных метода аутентификации для ограничения конкуренции;
- обеспечивает подключение к тысячам устройств с единой точкой доступа;
- включает возможность ретрансляции для уменьшения мощности на STA и позволяет использовать грубую форму mesh-сети с использованием метода однократного перехода;
- позволяет осуществлять расширенное управление питанием на каждом узле 802.11ah;
- позволяет общаться с незвездочной топологией с помощью Windows с ограниченным доступом;
- позволяет секторизацию, которая дает возможность группировать антенны для покрытия различных областей BSS (называемые секторами). Это достигается с помощью формирования диаграммы направленности, что привнесено из других протоколов 802.11.

Минимальная пропускная способность составит 150 кбит/с, это основано на модуляции BPSK на одном потоке MIMO с пропускной способностью 1 МГц. Максимальная теоретическая пропускная способность составит 347 Мбит/с на основе 256-WAM-модуляции с использованием 4 потоков MIMO и каналов 16 МГц.

Спецификация IEEE 802.11ah требует, чтобы STA поддерживали полосу пропускания 1 МГц и 2 ГГц. Точки доступа должны поддерживать каналы 1, 2 и 4 МГц. Каналы 8 МГц и 16 МГц являются дополнительными. Чем более узкая полоса пропускания канала, тем длиннее диапазон, но тем меньше пропускная способность. Чем больше пропускная способность канала, тем меньше диапазон, но тем это лучше для пропускной способности.

Ширина канала будет варьироваться в зависимости от региона, в котором развертывается 802.11ah. Некоторые комбинации не будут работать из-за правил в определенных регионах, как показано на рис. 6.21.

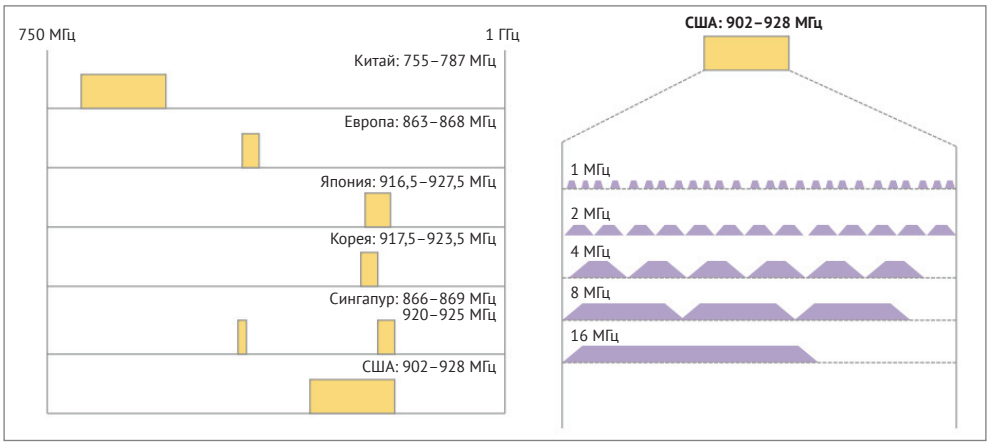


Рис. 6.21 ❖ Слева: различные варианты каналообразования в зависимости от региональных правил. Справа: различные варианты пропускной способности и канальная связь в регионе США с каналов 1 МГц до 16 МГц

Все в архитектуре стандарта IEEE 802.11ah направлено на оптимизацию общего диапазона и эффективности. Это касается даже длины заголовков MAC.

Подключение нескольких тысяч устройств к одной точке доступа также выполняется с использованием уникального идентификатора ассоциации (AID) в 13 бит. Это позволяет группировать STA на основе критериев (освещение в прихожей, выключатель освещения и т. д.). Это позволяет AP подключаться к более 8191 STA (802.11 может поддерживать только 2007 STA). Тем не менее, многие узлы способны вызывать огромное количество столкновений в канале. Несмотря на то, что количество подключенных CTC увеличилось, цель заключалась в сокращении объема данных в пути для передачи этим станциям. Целевая группа IEEE выполнила это, удалив несколько полей, которые не были особенно важны для случаев использования IoT, таких как поля QoS и DS. На рис. 6.22 показаны кадры нисходящей линии связи 802.11ah и восходящей линии связи по сравнению со стандартом 802.11.

Еще одно улучшение для управления питанием и эффективностью канала объясняется удалением фреймов подтверждения. ACK не явны для двунаправленных данных. То есть оба устройства отправляют и получают данные друг от друга. Обычно ACK будет использоваться после успешного приема пакета. В этом режиме bidi (BDT) прием следующего кадра предполагает, что предыдущие данные были успешно получены, и пакет ACK не нуждается в обмене.

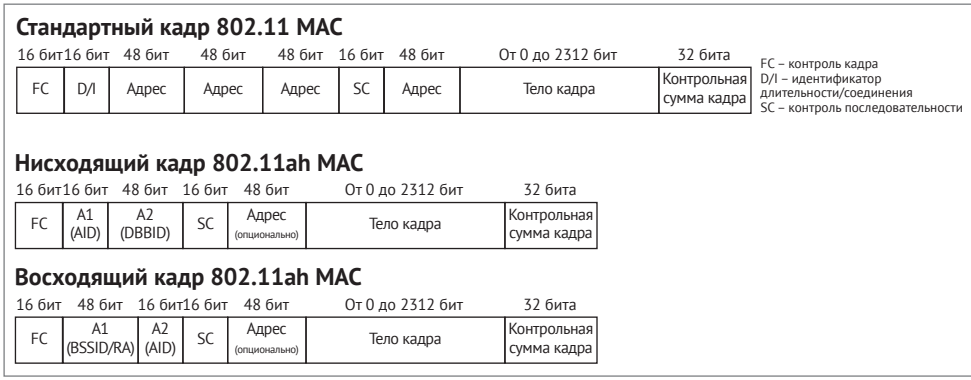


Рис. 6.22 ❖ Сравнение стандартного кадра 802.11 MAC и сжатого кадра 802.11ah

Чтобы избежать моря столкновений, которое помешало бы функциональности сети, 802.11ah использует **окно ограниченного доступа (RAW)**. Поскольку STA делятся на различные группы с использованием AID, каналы будут разделены на временные интервалы. Каждой группе назначается определенный временной интервал, а другие не назначаются. Существуют исключения, но для общего случая группировка образует некую изоляцию. Дополнительным преимуществом RAW является то, что устройства могут перейти в состояние гибернации для экономии энергии, если это не их временной интервал для передачи.

С точки зрения топологии, в сети 802.11ah есть три типа станций:

- **Root access point** – начало. Как правило, служит шлюзом для других сетей (WAN);
- **STA** – типичная станция 802.11 или конечный клиент;
- **Relaynode** – специальный узел, который объединяет AP-интерфейс с STA; находящийся в нижней BSS, и интерфейс STA для других узлов ретрансляции или корневой AP на верхней BSS.

На рис. 6.23 показана топология IEEE 802.11ah. Эта архитектура существенно отличается от других протоколов 802.11 при использовании единственных ретрансляционных узлов, которые работают для создания идентифицируемой BSS. Иерархия соотношений формирует большую сеть. Каждое соотношение действует как AP и STA.

В дополнение к основным типам узлов существуют три состояния энергосбережения, в которых может находиться STA:

- **карта индикации трафика (TIM)** – прослушивает AP для передачи данных. Узлы будут периодически получать информацию о данных, буферизованных для них из своей точки доступа. Отправленное сообщение называется информационным элементом TIM;
- **станции, не относящиеся к TIM,** – ведение переговоров с AP непосредственно во время связи для получения времени передачи в **окнах с ограниченным доступом (PRAW)**;

- **внеплановые станции** – не прослушивает никаких маяков и использует опрос для доступа к каналам.

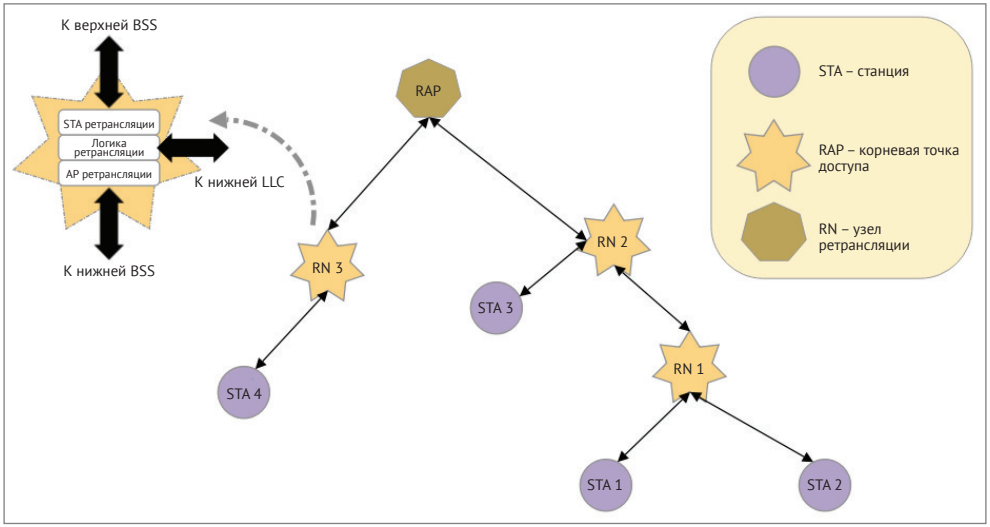


Рис. 6.23 ❖ Сетевая топология IEEE802.11ah

Энергопотребление имеет решающее значение для датчиков IoT и периферийных устройств на основе батарей. Протоколы 802.11 известны тем, что предъявляют высокие требования к мощности. Чтобы восстановить мощность этого беспроводного протокола, 802.11ah использует значение Max Idle Period, которое является частью обычных спецификаций 802.11. В общей сети 802.11 максимальный период ожидания составляет примерно 16 часов в зависимости от времени 16-битного разрешения. В 802.11ah первые два бита 16-разрядного таймера являются коэффициентом масштабирования, который позволяет длительность сна повысить до пяти лет.

Дополнительное энергопотребление смягчается посредством изменений в маяке. Как уже упоминалось ранее, маяки ретранслируют информацию о наличии буферизованных кадров. Маяки будут иметь битмапы TIM, которое увеличивает их размер, поскольку 8191 STA приведет к существенному увеличению битмапов. 802.11ah использует концепцию сегментации TIM, где некоторые маяки несут части общего битмапа. Каждая STA вычисляет, когда их соответствующий маяк с информацией о битмапе поступит и позволит устройству войти в режим энергосбережения и до момента, когда ему нужно проснуться и получить информацию о маяке.

Другая функция энергосбережения называется **целевым временем пробуждения (TWT)** и предназначена для STA, которые редко передают или полу-

чают данные. Это очень распространено в развертываниях IoT, таких как данные от датчика температуры. STA и связанная с ней AP будут вести переговоры, чтобы прийти к согласованной TWT, и STA войдет в состояние ожидания до тех пор, пока этот таймер не будет сигнализирован.

Процесс неявных ACK называется *Speed Frame Exchange*, что показано на рис. 6.24.

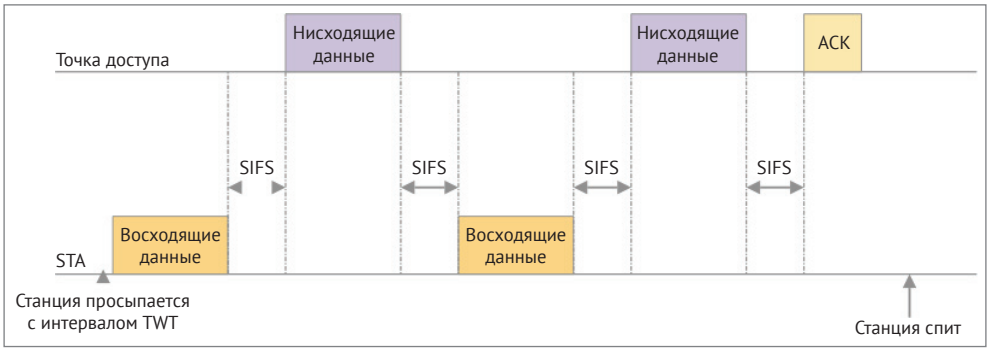


Рис. 6.24 ❖ Speed Frame Exchange 802.11ah: пример целевого времени пробуждения (TWT), используемый для запуска передачи STA. SIFS представляет собой разрыв между передачей AP и STA. Между парами данных не используются ACK. Только один ACK в конце передачи отправляется до того, как STA вернется в спящий режим

Заключение

В этой главе рассматривалась необходимая часть коммуникаций IoT. Использование стандартной связи на основе IP значительно упрощает дизайн и позволяет быстро и легко масштабировать работу. Масштабирование имеет решающее значение для развертываний IoT, которые охватывают тысячи или миллионы узлов. Использование транспорта на основе IP позволяет работать с помощью простых инструментов. 6LoWPAN и Thread демонстрируют стандарты, которые могут применяться к традиционно не IP-протоколам, таким как 802.15.4. Оба протокола позволяют адресовать IPv6 и mesh-сеть в массивные сети IoT. 802.11 является важным и чрезвычайно успешным протоколом, который составляет основу WLAN, но также может действовать в устройствах IoT и датчиках с использованием 802.11ah или транспортных системах с использованием 802.11p. Табл. 6.10 содержит сравнение традиционного протокола, отличного от IP, с протоколом IP. В основном, разница будет в мощности, скорости и диапазоне.

Таблица 6.10. Сравнение традиционного протокола, отличного от IP, с протоколом IP

	802.15.4	**802.11ah**
IP	Не-IP (требуется 6LoWPAN или Thread)	IP
Расстояние	100 м	До 1000 м
Структура сети	Полностью mesh	Иерархическая с одним переходом
Канализация	ISM 2,4 ГГц только с DSSS	Ниже 1 ГГц ISM с различными схемами кодирования модуляции. Полоса пропускания канала: 1, 2, 4, 8, 16 МГц
Управление интерференцией каналов	CSMA/CA	RAW, позволяющий STA связывать временные интервалы на основе групп
Пропускная способность	250 Кб/с	От 150 Кб/с до 347 Мб/с
Задержка	Хорошая	Лучшая (в 2 раза лучше, чем в 802.15.4)
Энергетическая эффективность	Наилучшая (17 мДж/пакет)	Хорошая (63 мДж/пакет)
Сохранение энергии	Механизмы сна-пробуждения в кадрах	Множество структур данных для управления и тонкой настройки мощности на разных уровнях
Размер сети	До 65 000	8192 STA

В следующей главе пойдет речь о протоколах очень дальнего диапазона и глобальной сети. То есть традиционные сотовые (4G LTE) и сотовые модели IoT, такие как Cat-1. В этой главе также будут обсуждаться протоколы LPWAN, такие как Sigfox и LoRa. WAN является следующим необходимым компонентом для получения данных в интернете.

Глава 7

Системы и протоколы дальней связи (ГВС)

На данный момент, нам удалось обсудить принцип работы **персональных сетей (WPAN)** и **локальных сетей (WLAN)**. В таких системах передачи данных датчики подключаются к локальной сети, но не обязательно к сети Интернет или другим системам. Необходимо помнить, что экосфера интернета вещей включает в себя датчики, исполнительные устройства, камеры, встроенные интеллектуальные устройства, транспортные средства и роботов, которые находятся в самых удаленных местах. Для передачи данных на большом расстоянии необходимо обращаться к **глобальной вычислительной сети (ГВС)**.

В этой главе охватываются основные устройства и топология глобальной сети, включая сотовую (4G и развивающуюся 5G) и другие проприетарные системы связи, такие как **LoRa и** Sighfox. Так как в этой главе рассматриваются системы сотовой и дальней связи с точки зрения данных, в ней не будет заостряться внимание на аналоговых и голосовых составляющих мобильных устройств. Дальняя передача данных – это обычно услуга, получаемая при подписке на поставщика, который располагает вышкой сотовой связи и предоставляет усовершенствованную инфраструктуру. Отличие предыдущих архитектур WPAN и WLAN заключается в том, что они существуют в рамках устройства, которое клиент или разработчик производит или перепродает. Подписка или соглашение об уровне предоставления услуги (SLA) имеет другое влияние на архитектуру и ограничения систем, в которых должен разбираться архитектор.

Функциональная совместимость устройств сотовой связи

Самый распространенный формат обмена данными – путем сотовой радиосвязи, где особое место занимает сотовая передача данных. Так как устройства мобильной связи существовали задолго до изобретения технологии сотовой связи, они имели ограниченную зону действия, общий диапазон частот и работали преимущественно по принципу двусторонней радиосвязи. Американская

корпорация Bell Labs (Лаборатории Белла) работала над созданием экспериментальных версий технологии мобильной связи в 1940-х гг. (Служба мобильной связи) и в 1950-х гг. (Усовершенствованная служба мобильной связи), но они имели незначительный успех. В то время еще не существовало универсальных стандартов для мобильной телефонной связи. Только когда концепция развития мобильной связи была разработана Дугласом Рингом и Реем Янгом в 1947 г. и реализована Ричардом Френкелем, Джоэлом Энгелем и Филипом Портером в офисе Bell Labs в 1960-х гг., можно было говорить о более широком и эффективном использовании мобильных технологий. Концепция «межсотового хэндовера» была создана и реализована Амосом Е. Джоэлом – младшим, также в офисе компании Bell Labs. Она подразумевала подключение абонента к новой базовой станции (соте) при перемещении мобильного устройства. Все эти технологии сложили воедино и создали первую систему сотовых телефонов, первый сотовый телефон; первый звонок по такому телефону совершил Мартин Купер, сотрудник компании Motorola, 3 апреля 1979 г.

Получилась идеальная ячеечная модель, в которой сотовые ячейки представлены шестиугольными участками с оптимальным размещением, показанная на рис. 7.1.

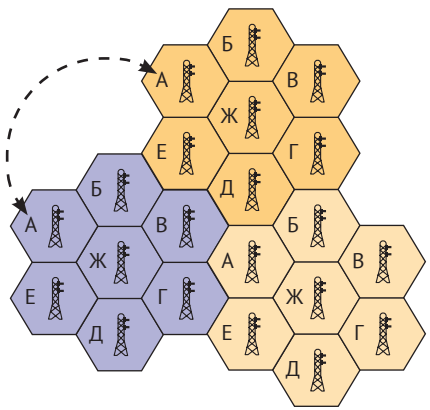


Рис. 7.1 ❖ Теория устройства сотовой сети.
Шестисторонняя модель расположения ячеек гарантирует разделение частот, избегая пересечения на одной и той же частоте ближайших соседей. На одном шестистороннем поле не может быть пересечения двух частот, как показано на примере ячейки А в двух разных регионах, что допускает повторное использование частоты

Разработка технологий и подтверждение действенности проектных решений в конечном итоге привели к развертыванию коммерческого производства и общественному признанию системы мобильных телефонов: в 1979 г. японской телекоммуникационной компанией Nippon Telegraph and Telephone (NTT), затем Данией, Финляндией, Норвегией и Швецией в 1981 г. В странах

Северной и Южной Америки система сотовой связи появилась лишь в 1983 г. Первые технологии сотовой связи известны как 1G или «первое поколение технологий сотовой связи» (1st generation). Далее будут детально рассмотрены основные особенности поколений сотовой связи, однако, следующий раздел уделит особое внимание описанию 4G LTE (эволюция 4G), так как это является современным стандартом мобильной связи и передачи данных. В последующих разделах будут описаны другие стандарты *интернета вещей* и будущие стандарты сотовой связи такие как NB-IOT (стандарт сотовой связи для устройств телеметрии с низкими объемами обмена данными) и 5G.

Стандарты и модель управления

Международный союз электросвязи (МСЭ) – это специализированное учреждение Организации Объединенных Наций, которое было основано в 1865 г. Оно приобрело свое современное название в 1932 г., еще до того, как стало специализированным учреждением ООН. Оно играет важную роль в установлении мировых стандартов беспроводной связи, навигации, мобильных технологий, обмена данными, интернета, голосовых данных и сетей следующего поколения. МСЭ включает 193 стран-участниц и 700 государственных и частных организаций. Организации поделены на рабочие группы, которые называются секторами.

Сектор, отвечающий за установку стандартов сотовой связи, называется **Сектор радиосвязи (МСЭ-Р). МСЭ-Р** – это орган, который определяет международные стандарты и цели для разных поколений радио- и сотовой связи, что включает установление стандартов надежности устройства и минимальной скорости передачи данных.
МСЭ-Р выпустили две основополагающие спецификации, которые обуславливали функционирование сотовой связи в прошлом десятилетии. Первая из них называется **Международные мобильные телекоммуникации-2000 (IMT-2000),** в ней устанавливаются требования для 3G устройств. Позднее, МСЭ-Р выпустили спецификацию под названием **Усовершенствованные международные мобильные телекоммуникации (IMT-Advanced).** Система **IMT-Advanced** основана на широкополосном беспроводном мобильном соединении с повсеместным использованием межсетевого протокола IP. IMT-Advanced определяет параметры соединения, которое может считаться соединением 4G во всем мире. Именно МСЭ-Р одобрили технологию **LTE** для дорожной карты 3GPP (партнерский проект третьего поколения), чтобы поддержать идею развития сотового соединения 4G в октябре 2010 г. МСЭ-Р продолжают разрабатывать новую систему требований для соединения 5G.

Примеры усовершенствованных требований МСЭ-Р для системы сотовой связи 4G включают:

- сеть с пакетной коммутацией, полностью на протоколе IP;
- совместимость с существующими беспроводными видами соединения;
- номинальная скорость передачи данных должна равняться 100 Мбит/с, когда клиент находится в движении, и 1 Гбит/с, когда клиент неподвижен;
- совместное динамичное использование сетевых ресурсов для поддержания более одного пользователя на ячейку;

- наращиваемая ширина/пропускаемость канала связи (от 5 до 20 МГц);
- безразрывная совместимость и глобальный роуминг внутри многоканальных сетей.

Проблема в том, что часто большое количество требований МСЭ-Р не выполняется и по этой причине бывает затруднительно дать название типу соединения (см. табл. 7.1).

Другой орган, занимающийся установкой стандартов в мире сотовой связи, – 3GPP (акроним от «Партнерский проект 3-го поколения») состоит из 7 телекоммуникационных организаций (также известных как «Партнерские организации») со всего мира, которые оказывают влияние на работу и развитие технологий сотовой связи. Был создан в 1998 г. в сотрудничестве с канадской компанией Nortel Networks и американской телекоммуникационной компанией AT&T Wireless и выпустил первый стандарт в 2000 г. Партнерские организации и Market Representatives работают в Японии, Америке, Китае, Европе, Индии и Корее. Общая цель группы – признание уже установленных стандартов и спецификаций для Глобальной системы мобильной связи (GSM) в процессе создания 3G спецификаций для сотовой связи. Обязанности 3GPP исполняются тремя **Группами технических спецификаций (TSG)**и шестью **Рабочими группами (WG).** Группы собираются несколько раз в год в разных регионах. Главной задачей релизов 3GPP является сделать нисходящую и восходящую совместимость устройств возможной.

В индустрии сотовой связи существует недопонимание различий между определениями МСЭ-Р, 3GPP и LTE. Давайте разберемся: МСЭ-Р определяет цели и стандарты для устройств, поддерживающих 4G и 5G. 3GPP отвечает за достижение целей в области таких технологий, как система модернизации LTE. МСЭ-Р занимается подтверждением того, что такие LTE продвижения соответствуют требованиям, установленным для 4G и 5G.

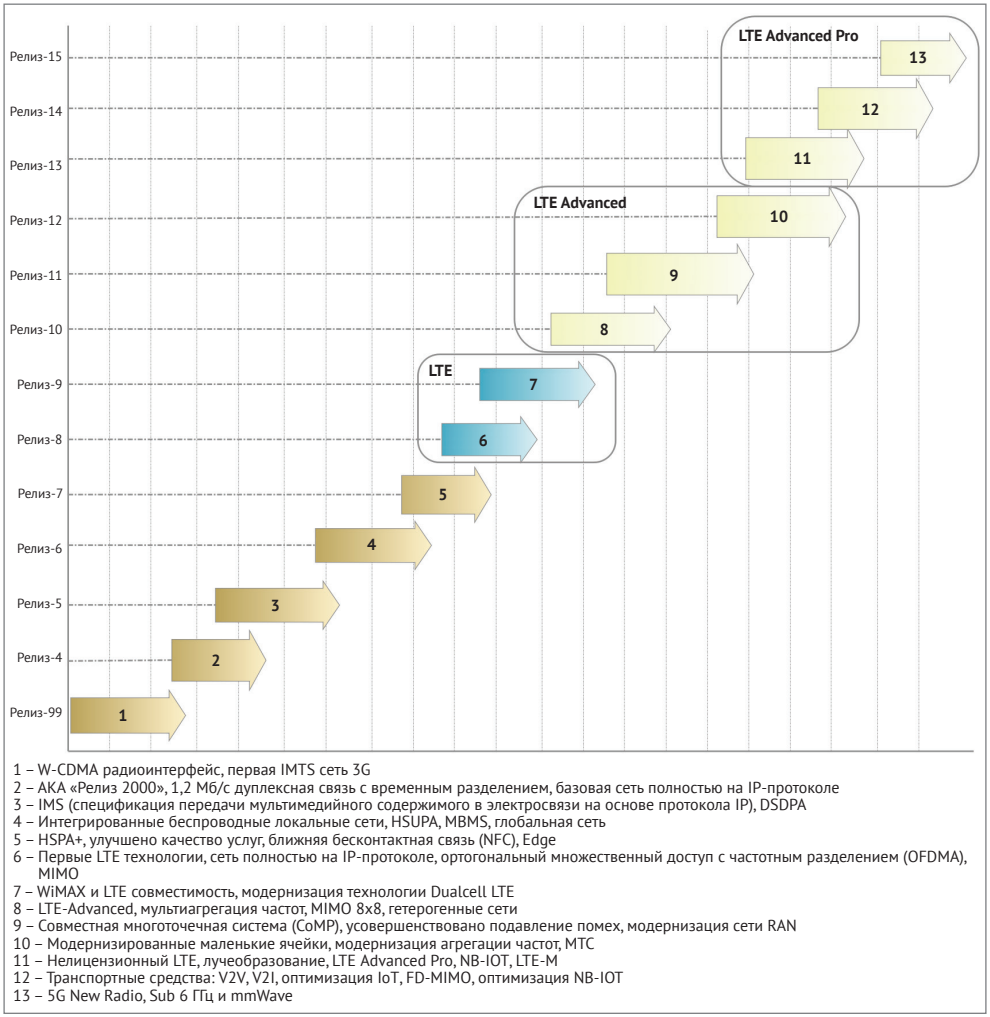
На рис. 7.2 показаны релизы технологии 3GPP с 2000 г. (в рамке показаны технологии LTE).

На LTE и его роль в профессиональном языке области сотовой связи часто приходится доля недопонимания. LTE значит Long-Term Evolution, многолетняя эволюция, она и является путем достижения установленных темпов и требований МСЭ-Р (которые изначально очень высоки). Продавцы мобильных телефонов выпускают смартфоны новых моделей в существующей зоне действия сотовой связи, используя устаревшие технологии, такие как 3G. Поставщики рекламируют 4G-LTE-совместимость, если они продемонстрировали существенные улучшения в скорости и характеристиках по сравнению с сетями 3G. С середины и до конца 2000-х гг. многим поставщикам не удавалось достичь полного соответствия требованиям спецификации МСЭ-Р для 4G, но они были очень близки. Поставщики использовали устаревшие технологии и выдавали их за 4G в большом количестве случаев. LTE Advanced – это еще одно улучшение, приближенное к целям МСЭ-Р.

Подводя итог, можно сказать, что терминология может быть очень запутанной и вводящей в заблуждение и архитектор не должен ограничиваться названием, чтобы понять суть технологии.

Таблица 7.1. Типы сотовых соединений

Тип соединения	1G	2/2,5G	3G	4G	5G
Время изобретения	1979	1999	2002	2010	2020
Спецификация МСЭ-Р	Нет сведений	Нет сведений	IMT-2000	IMT-Advanced	IMT-2020
Спецификация частоты МСЭ-Р	Нет сведений	Нет сведений	От 400 МГц до 3 ГГц	От 450 МГц до 3,6 ГГц	Неизвестно
Спецификация ширины/ пропускаемости канала связи	Нет сведений	Нет сведений	В движении: 348 Кбит/с. в уст. положении: 2 Мб/с	В движении: 100 Мб/с. В уст. положении: 1 Гбит/с	Min Down: 20 Гбит/с. Min Up: 10 Гбит/с
Стандартная ширина/пропускаемость канала связи	2 Кбит/с	14,4–64 Кбит/с	От 500 до 700 Кбит/с	От 100 до 300 Мб/с (максимум)	Неизвестно
Эксплуатация/Характеристики	Только мобильная телефония	Цифровая речь, SMS-сообщения автоматический определитель номера. Односторонняя передача данных	Более совершенное аудио, видео и передача данных. Расширена зона роуминга	Единый IP, Бесперебойная работа LAN/WAN/WLAN	Интернет вещей, сверхплотность каналов связи, низкое время ожидания подключения
Стандарты и мультиплексирование	Усовершенствованная служба мобильной связи (AMPS)	2G: TDMA,CDMA, GSM; 2,5G: GPRS, EDGE, 1xRTT	FDMA, TDMA, WCDMA, CDMA-2000	CDMA	CDMA
Хэндовер	Горизонтальный	Горизонтальный	Горизонтальный	Горизонтальный и вертикальный	Горизонтальный и вертикальный
Базовая сеть	ТСОП	ТСОП	С пакетной коммутацией	Интернет	Интернет
Коммутация	Соединение с коммутацией каналов	Соединение с коммутацией каналов в сети доступа и радиосети	Пакетная, за исключением радиоинтерфейса	Пакетная	Пакетная
Технология	Аналоговая Сотовая	Цифровая Сотовая	Широкополосная CDMA, WiMAX, IP-протокол	LTE Advanced Pro-based	LTE Advanced Pro-based mm Wave



1 – W-CDMA радиоинтерфейс, первая IMTS сеть 3G
2 – AKA «Релиз 2000», 1,2 Мб/с дуплексная связь с временным разделением, базовая сеть полностью на IP-протоколе
3 – IMS (спецификация передачи мультимедийного содержимого в электросвязи на основе протокола IP), DSDPA
4 – Интегрированные беспроводные локальные сети, HSUPA, MBMS, глобальная сеть
5 – HSPA+, улучшено качество услуг, ближняя бесконтактная связь (NFC), Edge
6 – Первые LTE технологии, сеть полностью на IP-протоколе, ортогональный множественный доступ с частотным разделением (OFDMA), MIMO
7 – WiMAX и LTE совместимость, модернизация технологии Dualcell LTE
8 – LTE-Advanced, мультиагрегация частот, MIMO 8x8, гетерогенные сети
9 – Совместная многоточечная система (CoMP), усовершенствовано подавление помех, модернизация сети RAN
10 – Модернизированные маленькие ячейки, модернизация агрегации частот, MTC
11 – Нелицензионный LTE, лучеобразование, LTE Advanced Pro, NB-IOT, LTE-M
12 – Транспортные средства: V2V, V2I, оптимизация IoT, FD-MIMO, оптимизация NB-IOT
13 – 5G New Radio, Sub 6 ГГц и mmWave

Рис. 7.2 ❖ Релизы 3GPP с 2000 по 2020 гг.

Технологии доступа сотовой связи

Важно понимать, как системы сотовой связи работают с многочисленными пользователями данных и голосовой связи. Существует несколько стандартов, схожих с концепциями, описанными для WPAN и WLAN, с которыми следует ознакомиться. До того, как у 3GPP была возможность поддерживать LTE, существовало множество стандартов для технологии сотовой связи, в частности, для устройств GSM и CDMA. Следует заметить, что эти технологии не совместимы друг с другом, начиная с инфраструктуры и заканчивая самим устройством:

- **множественный доступ с разделением частот (FDMA)** – распространен в аналоговых системах, но сейчас редко используется в цифровых доменах. Это техника, согласно которой спектр делится на частоты и распределяется среди пользователей. Один трансивер в любой данный промежуток времени приписывается к каналу связи. Канал, соответственно, закрыт для других звонков до тех пор, пока не завершится первоначальный звонок или пока он не будет подключен к другому каналу. Полностью дуплексный режим передачи требует наличия двух каналов связи: одного для передачи, другого для приема;
- **множественный доступ с кодовым разделением (CDMA)** – основан на технологии расширения спектра. CDMA увеличивает пропускную способность спектра, позволяя пользователям занять все каналы связи одновременно. Передача происходит по всему радиодиапазону, и каждому сигналу присваивается уникальный код для его вычленения из общей какофонии сигналов по всему спектру. CDMA допускает «мягкий хэндовер», что значит, что терминалы могут подключаться сразу к нескольким базовым станциям одновременно. Доминирующий радиоинтерфейс для мобильной связи третьего поколения был разработан американской компанией по разработке и исследованию беспроводных средств связи Qualcomm как CDMA2000 и предназначался для 3G. Из-за своей проприетарной природы он не был внедрен повсеместно и используется менее чем 18% мирового рынка. Был введен в США, где компании Verizon и Sprint являлись его постоянными поставщиками;
- **множественный доступ с временным разделением каналов (TDMA)** – в этой модели пропускная способность спектра увеличивается путем разделения каждой частоты на временные интервалы. TDMA позволяет каждому пользователю получать доступ ко всему радиочастотному каналу на краткий период звонка. Другие пользователи имеют доступ к этой же частоте в разные промежутки времени. Базовая станция постоянно переключается с пользователя на пользователя, находящегося на данном канале связи. TDMA является доминирующей технологией для мобильных сетей второго поколения. Организация GSM внедрила TDMA как модель множественного доступа. Она свойственна для 4 различных диапазонов частот: 900 МГц/1800 МГц в Европе и Азии и 850 МГц/1900 МГц в Северной и Южной Америке;
- некоторые устройства и модемы до сих пор поддерживают GSM/LTE или CDMA/LTE. GSM и CDMA несовместимы друг с другом. GSM/LTE и CDMA/LTE могут быть совместимы, однако, если они будут поддерживать диапазон частот LTE. В более старых моделях информация в звуковой форме передается при помощи спектра 2G или 3G, которые в значительной степени отличаются от CDMA и GSM (TDMA). Данные также несовместимы, так как данные LTE передаются на диапазонах частот 4G.

Категории абонентского оборудования 3GPP

В релизе 8 было 5 категорий абонентского оборудования, каждая с разной скоростью передачи данных и архитектурами MIMO (табл. 7.2). Категории позволили 3GPP различать между собой эволюции LTE. После 8-го релиза было добавлено больше категорий. В категориях совмещаются возможности приема и передачи данных, как указано организацией 3GPP. Как правило, вы получаете сотовое радио или чипсет, на которых указано, какую категорию они поддерживают. Если оборудование пользователя поддерживает определенную категорию, сотовая система (eNodeB, о которой будет говориться далее) также должна поддерживать эту категорию.

Частью процесса ассоциации между устройством сотовой связи и инфраструктурой является обмен информацией о возможностях, таких как категория.

Таблица 7.2. Категории абонентского оборудования 3GPP

Релиз 3GPP	Категория	Максимальная скорость передачи данных при нисходящей связи (Мб/с) L1	Максимальная скорость передачи данных при восходящей связи (Мб/с) L1	Максимальное количество каналов нисходящей передачи данных MIMO
8	5	299,6	75,4	4
8	4	150,8	51	2
8	3	102	51	2
8	2	51	25,5	2
8	1	10,3	5,2	1
10	8	2998,60	1497,80	8
10	7	301,5	102	2 или 4
10	6	301,5	51	2 или 4
11	9	452,2	51	2 или 4
11	12	603	102	2 или 4
11	11	603	51	2 или 4
11	10	452,2	102	2 или 4
12	16	979	неизвестно	2 или 4
12	15	750	226	2 или 4
12	14	3,917	9,585	8
12	13	391,7	150,8	2 или 4
12	0	1	1	1
13	NB1	0,68	1	1
13	M1	1	1	1
13	19	1566	неизвестно	2,4 или 8
13	18	1174	неизвестно	2,4 или 8
13	17	25,065	неизвестно	8

Обратите внимание на категории M1 и MB1 в релизе 13. Здесь организация 3GPP значительно уменьшила скорость передачи данных до 1 Мб/с и менее. Это категории, предназначенные для устройств с функцией интернет вещей, которые требуют низкую скорость передачи данных, так как соединение происходит посредством посыла кратких сигналов.

Распределение спектра и полос частот в 4G LTE

Существует 55 диапазонов частот LTE, на что частично повлияла фрагментация спектра и рыночные стратегии. Распределение полос частот LTE также является заслугой аукциона частот, который проводится государством. LTE также делится на 2 категории, которые несовместимы друг с другом:

- **дуплекс с временным разделением (TDD) –** TDD использует общий диапазон частот для передачи и приема данных. Направление передачи определяется временными слотами;
- **дуплекс с частотным разделением (FDD) –** в конфигурации FDD базовая станция (eNodeB) и **абонентское устройство (UE)** открывают два диапазона частот для приема и передачи данных. Примером этому может быть полоса частот LTE-13, у которой диапазон передачи данных варьируется от 777 до 787 МГц, а диапазон приема данных от 746 до 756 МГц. Передача и прием данных могут происходить одновременно.

Существует режим комбинирования модулей TDD и FDD, который объединяет их в одном модеме и допускает использование нескольких несущих.

Обратите внимание на то, что принимаемые данные будут относиться к типу связи «от базовой станции до устройства абонента», а передаваемые данные будут иметь противоположное направление.

Некоторые другие термины из LTE-сферы также важны для понимания распределения спектра:

- **ресурсный элемент –** минимальный блок передачи данных в LTE. Он состоит из одной поднесущей на единицу времени передачи символов (OFDM или SC-FDM);
- **разнос поднесущих:** это расстояние между поднесущими. LTE использует 15 кГц без защитных полос;
- **циклический префикс –** так как нет защитных полос, используется циклический префикс для предотвращения многолучевой межсимвольной интерференции между поднесущими;
- **временной слот –** LTE использует временной слот длиною в 0,5 мс для кадров LTE. Он равняется 6–7 символам OFDM в зависимости от длительности циклического префикса;
- **ресурсный блок:** один блок передачи данных. Содержит 12 поднесущих и 7 символов, что равняется 84 ресурсным элементам.

Кадр LTE длиной в 10 мс состоит из 10 подкадров (рис. 7.3). Если 10% общей ширины полосы частот канала 20 МГц используется для циклического префикса, то эффективная полоса частот сужается до 18 МГц. Количество поднесущих в 18 МГц: 18 МГц/15 МГц = 1200. Количество ресурсных блоков: 18 МГц/180 кГц = 1000.

Полосы, выделенные для 4G LTE, различаются в зависимости от региональных норм (Северная Америка, Азиатско-Тихоокеанский регион и т. д.). Для каждой полосы есть ряд стандартов, разработанных и выпущенных 3GPP

и МСЭ-Р. Полосы разделены между зонами FDD и TDD и носят общепринятые названия-акронимы, которые регулярно используются в индустрии, такие как **Усовершенствованная беспроводная служба (AWS).** В табл. 7.3 показано разделение полос FDD и TDD только для североамериканского региона.

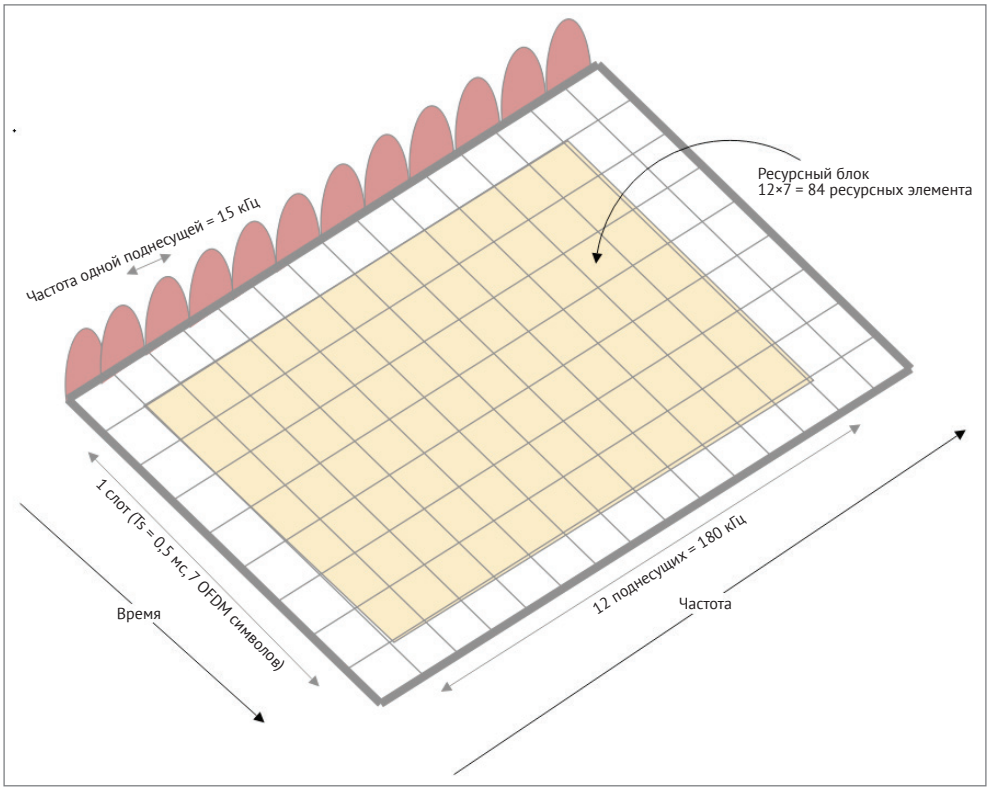


Рис. 7.3 ❖ Один слот кадра LTE. Кадр LTE длиной в 10 мс состоит из 20 слотов. Слоты основаны на 12 поднесущих шириной в 15 кГц и семи OFDM-символах, что, в общем, дает (12×7) 84 ресурсных элемента

Таблица 7.3. Распределение полос частот LTE 4G FDD и поставщики Северной Америки.

Полоса	Дуплекс	Частота (МГц)	Общее название	Северная Америка	Ширина полосы (МГц)	Дуплексный интервал (МГц)	Бандгап (МГц)
1	FDD	2100	IMT		60	190	130
2	FDD	1900	PCS блоки A-F	да	60	80	20
3	FDD	1800	DCS		75	95	20
4	FDD	1700	AWS блоки A-F (AWS1)	да	45	400	355
5	FDD	850	CLR	США (AT&T, U.S. Cellular)	25	45	20

Таблица 7.3 (продолжение)

Полоса	Дуплекс	Частота (МГц)	Общее название	Северная Америка	Ширина полосы (МГц)	Дуплексный интервал (МГц)	Бандгап (МГц)
6	FDD				10	35	25
7	FDD	2600	IMT-E	Канада (Bell, Rogers, Telus)	70	120	50
8	FDD	900	E-GSM		35	45	10
9	FDD				35	95	60
10	FDD	1700	Усов. AWS блоки A1		60	400	340
11	FDD	1500	Lower PDC	Канада (Bell), о. Гуам (iConnect,…)	20	48	28
12	FDD	700	Lower SMH блоки A/B/C	США (Verizon), Канада (Bell, EastLink,…)	18	30	12
13	FDD	700	Upper SMH блок C	США (First Net)	10	–31	41
14	FDD	700	Upper SMH блок D		10	–30	40
15	FDD	2000			20	700	680
16	FDD	700			15	575	560
17	FDD	700	Lower SMH блоки B/C	Канада (Rogers), о. Гуам (NTT), США (AT&T)	12	30	15
18	FDD	850	Japan Lower800		15	45	30
19	FDD	850	Japan Upper 800		15	45	30
20	FDD	800	Цифровой дивиденд ЕС		30	–41	71
21	FDD	1500	Upper PDC		15	48	33
22	FDD	3500		США (Ligado Networks)	90	100	10
23	FDD	2000			20	180	160
24	FDD	1600	L-band (США)		34	–101,5	135,5
25	FDD	1900	Усов. PCS блоки A-G	США (Sprint)	65	80	15
26	FDD	850	Усов. CLR	США (Sprint)	30/40		10
27	FDD	800	SMR		17	45	28
28	FDD	700	APT		45	55	10
29	FDD (A1)	700	Lower SMH блоки D/E	США (AT&T)	11	неизвестно	
30	FDD	2300	WCS блоки A/B	США (AT&T)	10	45	35
31	FDD	450			5	10	5
32	FDD (A1)	1500	L-band (EC)		44	неизв.	
65	FDD	2100	Усов. IMT		90	190	
66	FDD	1700	Усов. AWS Блоки A-J	Канада (Freedom Mobile)	90/70	400	

Таблица 7.3 (окончание)

Полоса	Дуплекс	Частота (МГц)	Общее название	Северная Америка	Ширина полосы (МГц)	Дуплексный интервал (МГц)	Бандгап (МГц)
67	FDD (A1)	700	EU 700		20	неизв.	
68	FDD	700	ME 700		30	55	
69	FDD (A1)	2600	IMT-E (дуплексный разнос)		50	неизв.	
70	FDD	2000	AWS-4	США (DISH)	25/15	300	
71	FDD	600	Цифровой дивиденд EC	США (T-Mobile)			

Таблица 7.4. Распределение полос частот LTE 4G FDD и поставщики Северной Америки

Полоса	Дуплекс	Частота (МГц)	Общее название	Северная Америка	Распределение (МГц)	Ширина полосы
33	TDD	2100	IMT		1900–1920	20
34	TDD	2100	IMT		2010–2025	15
35	TDD	1900	PCS (Uplink)		1850–1910	60
36	TDD	1900	PCS (Downlink)		1930–1990	60
37	TDD	1900	PCS (Duplex spacing)		1910–1930	20
38	TDD	2600	IMT-E (Duplex spacing)		2570–2620	50
39	TDD	1900	DCS-IMT gap		1880–1920	40
40	TDD	2300			2300–2400	100
41	TDD	2500	BRS/EBS	USA (Sprint)	2496–2690	194
42	TDD	3500			3400–3600	200
43	TDD	3700			3600–3800	200
44	TDD	700	APT		703–803	100
45	TDD	1500	L-Band (Китай)		1447–1467	20
46	TDD	5200	U-NII		5150–5295 (нелицензионное)	775
47	TDD	5900	U-NII-4 (V2x)		5855–5925 (нелицензионное)	70
48	TDD	3600	CBRS		3550–3700	150

LTE также был разработан с целью эксплуатации в нелицензионном спектре. Являясь идеей компании Qualcomm, изначально он функционировал на полосе шириной 5 ГГц согласно стандарту IEEE 802.11a. Суть идеи заключалась в том, чтобы он служил альтернативой точкам доступа Wi-Fi. Эта полоса частот шириной от 5150 до 5350 МГц обычно требует, чтобы радио работали на мощности 200 мВт (максимум) и только внутри помещений. На сегодняшний день только T-Mobile поддерживает использование LTE в районах США. AT&T и Verizon проводят открытое тестирование технологии LAA. Это три категории нелицензионного использования спектра для сотовой связи:

- **нелицензионный LTE (LTE-U) –** как уже упоминалось, он предполагает его сосуществование с Wi-Fi устройствами на полосе 5 ГГц. Контрольный канал LTE останется неизменным, в то время как голос и данные перейдут на полосу 5 ГГц. Суть концепции LTE-U заключается в том, что абонентское устройство может поддерживать только односторонний прием данных на нелицензионной полосе или при полном дуплексе;
- **Licensed-Assisted Access (LAA) –** эта технология схожа с LTE-U, но контролируется и разрабатывается организацией 3GPP. Она использует протокол под названием **«listen before talk» («послушай, прежде чем сказать» (LBT)**, чтобы облегчить сосуществование с Wi-Fi.

Новая нелицензионная технология под названием Multifire является еще одной возможностью использования нелицензионного спектра. Multifire – это разновидность LTE на полосе 5 ГГц как LTE-U и LAA. Однако Multifire не требуется лицензионная полоса для работы. Это подразумевает то, что ни один коммерческий поставщик (такой как AT&T или Verizon) не сможет предоставить такой тип связи. Если поставщики будут вынуждены покинуть производственно-сбытовую цепь, экономика изменится для передачи и владения данными, так как большая компания может создать свою собственную сотовую сеть и управлять ей, используя только нелицензионные частоты.

Multifire использует такую же LBT-технологию сосуществования, как LAA. В действительности, Multifire не будет нуждаться в поле сотовых ячеек для того, чтобы предоставить обслуживание (равно, как и точки доступа Wi-Fi). Еще одно преимущество Multifire в том, что полоса глобально используется соединением Wi-Fi и обеспечивает согласованность между регионами.

Топология и архитектура сети 4G LTE

Архитектура 3GPP LTE называется «**Эволюция системной архитектуры» (SEA),** и ее общей целью является предоставить упрощенную архитектуру, целиком построенную на IP. Она способна поддерживать высокоскоростную передачу данных с низкой задержкой в **сетях с радиодоступом (RANs).** В 8 релизе 3GPP roadmap был введен стандарт LTE. Такие сети состоят из компонентов с пакетной коммутацией на базе протокола IP, что значит, что голосовые данные отправляются в виде цифровых IP-пакетов. Это является еще одним коренным отличием от традиционной сети 3G.

В топологии 3G использовалась коммутация каналов для трафика речи и СМС и пакетная коммутация для данных. Коммутация каналов в корне отличается от пакетной коммутации. Коммутация каналов берет начало в оригинальной телефонной коммутированной сети. Она использует выделенный канал связи, который представляет собой соединение между узлом-адресантом и узлом-адресатом, это соединение на протяжении всего сеанса обмена информацией может использоваться только двумя указанными узлами. В сети с пакетной коммутацией сообщения разбиваются на небольшие части (которые называются пакетами в случае данных IP) и находят самый короткий путь от адресанта данных к их адресату. В заголовке, которым оснащен пакет, обязательно указан адрес узла-получателя.

Типичная 4G-LTE-сеть состоит из трех компонентов: клиент, радиосеть и базовая сеть. Клиентом является радиоустройство пользователя. Радиосеть представляет прямую связь между базовой сетью и клиентом и включает радиооборудование, такое как вышка. Базовая сеть представляет интерфейс управления поставщика и может поддерживать одну или более радиосетей.

Компонентами этой архитектуры являются:

- **сеть с расширенным универсальным наземным радиодоступом (EUTRAN) –** это радиоинтерфейс 4G LTE для устройств абонентского оборудования. EUTRAN использует OFDMA для нисходящей линии связи, SC-FDMA для восходящей линии связи с целью экономии энергии. Следовательно, это делает ее несовместимой с традиционной технологией 3GW-CDMA. Сеть состоит из узлов eNodeB, но может содержать несколько узлов, которые взаимодействуют между собой через канал, который называется интерфейс **X2;**
- **eNodeB –** это ядро радиосети. Он отвечает за сообщение между оборудованием абонента и усовершенствованным пакетным ядром (EPC). Каждый eNodeB – это базовая станция, которая контролирует абонентское оборудование в одной или более сотовых зонах и присваивает ресурсы определенному клиенту частями, размером в 1 мс, которые называются **интервалами передачи (TTI).** Он распределяет ресурсы канала связи между различными UE на основании условий эксплуатации. Системы eNodeB также отвечают за изменение состояния с «не занято» до «подключено». Он также обеспечивает мобильность абонентского оборудования в лице хэндовера для подключения к другим eNodeB. Кроме того, он контролирует передачу данных и отслеживает перегрузки сети. Интерфейс взаимодействия eNodeB и EPC – интерфейс S1;
- **абонентское оборудование (UE) –** это аппаратное обеспечение клиента (хардвер), состоящее из **абонентских устройств**, которые выполняют все функции связи **оконечного/терминального оборудования**, преобразующего потоки информации в данные и **универсальной карты с интегральной схемой (UICC),** то есть SIM-карта, которая служит для идентификации пользователя;
- **усовершенствованное пакетное ядро (EPC):** при создании LTE организация 3GPP решила создать плоскую архитектуру и отделить данные пользователя (плоскость пользователя) отданных управления (плоскость управления), что обеспечивает более эффективное масштабирование. EPC состоит из 5 основных компонентов:
  - **узел управления мобильностью (MME**) – отвечает за трафик плоскости управления, аутентификацию и безопасность пользователя, слежение и пейджинг, а также за процедуры обеспечения мобильности. MME должен распознавать мобильность в неактивном режиме. Это достигается при помощи кода **Зоны отслеживания (TA).** MME также контролирует сигнализацию **Слоя без доступа (NAS)** и контроль переноса информации (о котором будет говорится далее);

- **опорный абонентский сервер (HSS) –** представляет собой большую базу данных, связанную с MME, и предназначен для хранения данных об абонентах, что может включать: ключи, данные пользователя, максимальную скорость передачи данных на тарифном плане, подписки и т. д.;
- **обслуживающий шлюз (SGW) –** отвечает за плоскость пользователя и поток данных пользователя. Главным образом он служит в качестве маршрутизатора и направляет пакеты с пользовательскими данными непосредственно между eNodeB и PGW. Интерфейс, приуроченный к SGW, называется S5/S8. S5 используется, если два устройства находятся в одной и той же сети, а S8 – когда устройства находятся в разных сетях;
- **шлюз сети общего доступа (PGW) –** обеспечивает соединение от UE к внешним сетям данных, включая интернет и другие сети общего доступа. Также он распределяет IP-адреса для подключенных мобильных устройств. PGW занимается вопросом качества обслуживания, таким как трансляция видео и просмотр веб-страниц. Он использует интерфейс под названием SGi для соединения с различными внешними сетями;
- **узел выставления счетов абонентам (PCRF) –** это еще одна база данных, в которой хранится информация о политиках и правилах принятия решений. Он также контролирует функции взимания оплаты;

❍ **сеть общего доступа (PDN) –** является внешним интерфейсом и, в большинстве случаев, сетью Интернет. Может включать другие услуги, центр регистрации и обработки данных, частные услуги и т. д.

В услуге 4G LTE абоненту предоставляется поставщик или оператор, известный как **Сеть связи общего пользования наземных мобильных объектов (PLMN).** Если абонент находится в сети PLMN своего поставщика, то он или она находится в домашней PLMN-сети. Если же абонент перемещается в другую сеть, которая находится за пределами его домашней сети (например, во время поездок за границу), она называется **гостевой PLMN.** Абонент подключает свое оборудование к гостевой сети, что требует ресурсов E-UTRAN, MME, SGW и PGW новой сети. PGW предоставляет доступ к интернету. В этот момент услуги роуминга начинают влиять на тарифный план. Оплата за роуминг взимается гостевой сетью и указывается в счете клиента. Ниже вы увидите иллюстрацию к этой архитектуре. Справа изображена модель перемещения абонента в гостевую сеть и распределение функциональных возможностей между гостевой сетью E-UTRAN и усовершенствованным пакетным ядром (EPC), а также показан путь к домашней сети. Интерфейс S5 используется, если клиент и поставщик находятся в одной сети, а S8 – когда клиент пересекает границы других сетей (рис. 7.4).

Сигнализация вне уровня доступа (NAS) была упомянута в описании функций MME. Это механизм, который служит для передачи информации между UE и базовыми станциями, такими как центр мобильной коммутации.

Примером могут послужить такие пласты информации, как аутентификационные сообщения, обновления или сообщения о подключении. NAS находится на вершине стека протоколов SAE.

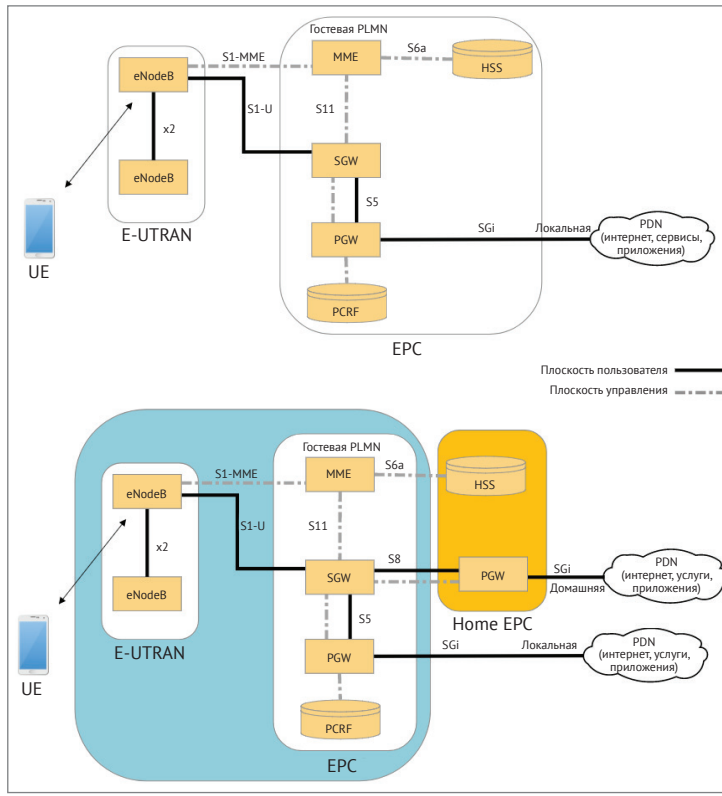


Рис. 7.4 ❖ Внизу: архитектура системы 3GPP. Вверху: архитектура 4G LTE в общих чертах

Протокол GTP (GPRS Tunneling Protocol) – это протокол, основанный на IP/UPD (протокол пользовательских датаграмм) в сети LTE. Он используется в коммуникационной инфраструктуре LTE для данных управления, данных пользователя и данных о стоимости связи. На схеме, представленной на рис. 7.4, большинство из S* компонентов, объединяющих каналы, используют пакеты GTP.

Архитектура LTE и стек протоколов используют так называемые **носители.** Носители являются виртуальной концепцией, использующейся для предоставления «канала» для передачи данных от одного узла к другому. Канал между PGW и UE называется **EPS носитель.** Когда данные из интернета попадают в PGW, носитель преобразует данные в пакет GTP-U и отправляет их к SGW.

SGW получает пакет, стирает заголовок GTP-U и переупаковывает данные пользователя в новый GTP-U-пакет для отправки на базовую станцию. На базовой станции процесс повторяется, данные переупаковываются после сжатия, шифрования и отправки к логическим каналам связи. Данные передаются к абонентскому оборудованию через радионоситель. Одно из преимуществ, которые носители предоставляют LTE, – это контроль качества обслуживания. При использовании носителей инфраструктура может гарантировать определенную скорость передачи данных в зависимости от клиента, приложения или целей использования.

Когда абонентское оборудование подключается к сети сотовой связи впервые, ему присваивается *носитель по умолчанию*. У каждого такого носителя есть свой IP-адрес, и абонентское оборудование может иметь несколько стандартных носителей с уникальным IP. *Выделенный носитель*, в этом случае, используется для QoS и качественного взаимодействия с пользователем. Он вводится, когда носитель по умолчанию больше не способен предоставлять обслуживание. Выделенный носитель всегда находится выше носителя по умолчанию. У типичного смартфона одновременно могут функционировать следующие носители:

- носитель по умолчанию 1 – отправка сообщений, сигнализация протокола SIP;
- выделенный носитель – голосовые данные (VOIP), соединенные с носителем 1;
- носитель по умолчанию 2 – все службы передачи данных, такие как email и браузер.

Такой аспект сети LTE, как коммутация, также заслуживает внимания. Мы рассмотрели разные поколения эволюций 3GPP от 1G до 5G. Одной из целей организации 3GPP и поставщиков было приблизиться к стандартному и общепринятому решению под названием **передача речи по протоколу IP (VOIP)**. Через стандартные IP-интерфейсы будут передаваться не только данные, но и голос. После некоторого соревнования между методами, организации, устанавливающие стандарты, сошлись на использовании архитектуры VoLTE. Архитектура VoLTE использует усовершенствованный вариант **протокола установления сеанса (SIP)** для работы с голосовыми и текстовыми сообщениями. Кодек, известный как **адаптивный многоскоростной (AMR),** предоставляет широкополосную высококачественную голосовую и видеосвязь. Далее мы рассмотрим новые категории 3GPP LTE, которые исключают VoLTE, чтобы поддерживать использование *интернета вещей*.

Необходимо отметить, что LTE является одним из двух стандартов для широкополосной мобильной связи. **Беспроводной доступ к мобильному интернету (WiMAX)** является его альтернативой. LTE WiMAX – это широкополосный, основанный на IP и OFDMA протокол связи. WiMAX основан на стандарте IEEE 802.16 и управляется Форумом WiMAX. WiMAX использует частоты 2,3 и 3,5 ГГц, но также может достигать 2,1 ГГц и 2,5 ГГц, подобно LTE. Будучи введенным на рынок ранее LTE, WiMAX является выбором компаний Sprint и Clearwire для высокоскоростной передачи данных.

Однако WiMAX нашел лишь узкопрофильное применение. LTE, как правило, намного более гибок и, следовательно, более широко используется. LTE также шаг за шагом развивался, чтобы добиться совместимости с более ранними инфраструктурами и технологиями, функционирующими по сей день, в то время как WiMAX предназначался для последующих изобретений. WiMAX превосходит LTE в плане легкости установки, но LTE выиграл «гонку широкополосности», которая, в свою очередь, определила революцию в сфере мобильной связи.

Стек протоколов сети E-UTRAN 4G LTE

Стек протоколов 4G LTE имеет схожие характеристики с другими производными сетевой модели OSI, однако также присутствуют различия в характеристиках плоскости управления 4G, как указано на рис. 7.5. Одно из различий заключается в том, что **Управление радиоресурсами (RRC)** распространяется на существенное количество уровней стека. Это называется **плоскостью управления.** Плоскость управления имеет два режима: *не занято* и *подключено*. В неподключенном состоянии абонентское оборудование ожидает в сотовой ячейке после соединения с ней и может следить за пейджинговым каналом, чтобы определить, предназначается ли входящий звонок или системная информация для него. В подключенном состоянии абонентское оборудование создает канал нисходящей связи и извлекает информацию о соседней ячейке. Сеть E-UTRAN использует эту информацию, чтобы найти наиболее подходящую ячейку на данный момент.

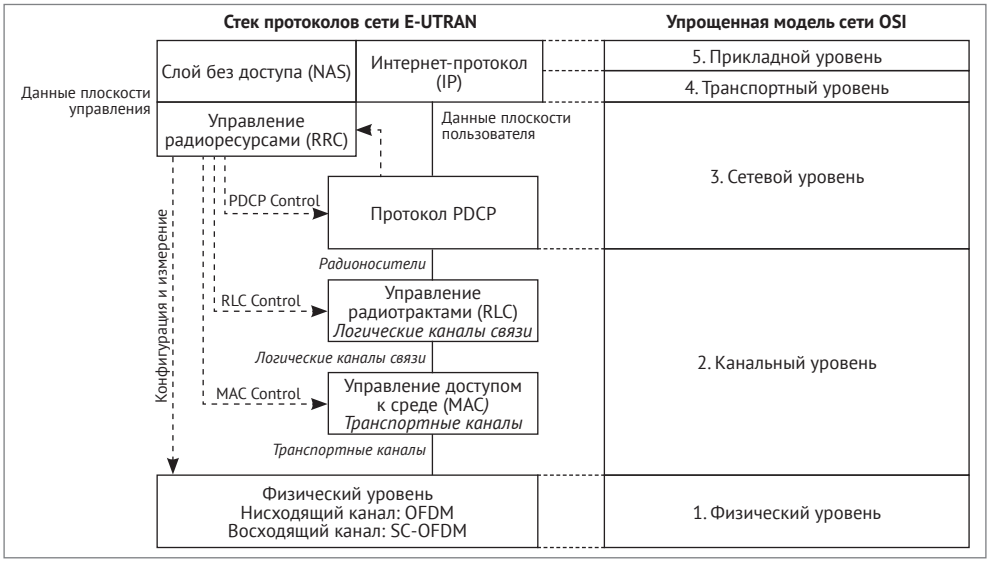


Рис. 7.5 ❖ Стек протоколов сети E-UTRAN для 4G LTE и сравнение с упрощенной моделью сети OSI

Плоскость управления и плоскость пользователя также функционируют отлично друг от друга и имеют разное время задержки/отклика. Плоскость пользователя обычно имеет задержку в 4,9 мс, в то время как плоскость управления обладает задержкой в 50 мс.

Стек протоколов состоит из следующих уровней и имеет такие характеристики:

- **физический уровень 1** – этот уровень является радиоинтерфейсом, который также известен как *воздушный интерфейс*. Имеет следующие обязанности: адаптация канала (AMC), управление мощностью, модуляция сигнала (OFDM), цифровая обработка сигналов, поиск сотовых сигналов, синхронизация с ячейками, управление хэндовером, измерение ячеек для уровня RRC;
- **управление доступом к среде (MAC)** – подобно другим стекам, основанным на OSI, выполняется сопоставление между логическими каналами и транспортными уровнями. Уровень MAC мультиплексирует различные пакеты в **транспортные блоки (ТБ)** для физического уровня. Другие обязанности включают планирование отчетов, исправление ошибок, приоритизацию каналов и управление несколькими UE;
- **управление радиоканалом (RLC)** – RLC передает PDU верхнего уровня, выполняет коррекцию ошибок через ARQ и выполняет конкатенацию/сегментацию пакетов. Кроме того, он обеспечивает интерфейс логического канала, обнаружение дублированных пакетов и выполняет повторную сборку полученных данных;
- **протокол управления конвергенцией пакетов (PDCP)** – этот уровень отвечает за сжатие и декомпрессию пакетов. PDCP также управляет маршрутизацией данных пользовательской плоскости, а также данными плоскости управления с обратной связью в RRC. Управление дублирующимися SDU (например, в процедуре передачи обслуживания) обрабатывается на этом уровне. Другие функции включают шифрование и дешифрование, защиту целостности, отбрасывание данных на основе таймера и восстановление каналов;
- **управление радиоресурсами (RRC)** – уровень RRC транслирует системную информацию на все уровни, включая **Слои без доступа (NAS)** и **Слои доступа (AS)**. Он управляет ключами безопасности, конфигурациями и управлением радионесущими;
- **слои без доступа** – это самый высокий уровень плоскости управления, он является основным интерфейсом между UE и MME. Основная роль – управление сеансом, поэтому мобильность UE устанавливается на этом уровне;
- **слой доступа**: это слой ниже NAS, целью которого является передача сигналов, отличных от радио, между UE и радиосетью.

Географические области 4G LTE, потоки данных и процедуры передачи обслуживания

Прежде чем рассматривать процесс передачи сотового обслуживания, мы должны сначала определить местные параметры и идентификацию сети. В сети LTE существует три типа географических областей:

- **области пула MME** – определяется как область, где UE может перемещаться без изменения обслуживающего MME;
- **области обслуживания SGW** – определяется как область, в которой один или несколько SGW будут продолжать предоставлять обслуживание для UE;
- **области отслеживания (TA)** – они определяют подзоны, состоящие из небольших областей MME и SGW, которые не перекрываются. Используются для отслеживания местоположения UE, находящегося в режиме ожидания. Важно для передачи обслуживания.

Каждая сеть в сети 4G LTE должна быть однозначно идентифицирована для работы этих служб. Чтобы помочь в определении сетей, 3GPP использует сетевой идентификатор, состоящий из:

- **мобильного кода страны (MCC)** – трехзначный идентификатор страны, в которой находится сеть (например, для Канады – 302);
- **кода мобильной сети (MNC)** – двузначное или трехзначное значение, указывающее несущую (например, для Rogers Wireless – 720).

Каждой несущей также необходимо однозначно идентифицировать каждый MME, который она использует и поддерживает. MME необходим локально в пределах одной сети и во всем мире, когда устройство передвигается и находится в роуминге, чтобы найти домашнюю сеть. Каждый MME имеет три идентификатора:

- **идентификатор MME** – это уникальный идентификатор, который будет определять конкретный MME в сети. Он состоит из следующих двух полей:
  - **MME-код (MMEC)** – идентифицирует все MME, принадлежащие к той же области пула, о которой упоминалось ранее;
  - **идентификация группы MME (MMEI)** – определяет группу или кластер MME;
- **глобальный уникальный идентификатор MME (GUMMEI)** – это комбинация PLNM-ID и MMEI, описанных ранее. Комбинация формирует идентификатор, который может быть расположен в любой точке мира в любой сети.

Идентификация области отслеживания (TAI) позволяет отслеживать устройство UE глобально из любой позиции. Это комбинация PLMN-ID и **кода зоны отслеживания (TAC)**. TAC представляет собой конкретную физическую подобласть зоны покрытия ячейки.

Идентификатор соты сконструирован из комбинации идентификатора соты **E-UTRAM (ECI)**, который идентифицирует ячейку в сети, глобального идентификатора соты **E-UTRAN (ECGI)**, который идентифицирует соту в любой точке мира, и физического идентификатора соты (целочисленное значение от 0 до 503), используемого для того, чтобы отличить ее от соседней EU.

Процесс передачи обслуживания включает в себя передачу сеанса вызова или данных с одного канала на другой в сотовой сети. Передача обслуживания

наиболее часто происходит, если клиент движется. Передача обслуживания также может быть инициирована, если базовая станция достигла предела своей пропускной способности и вынуждена переместить некоторые устройства на другую базовую станцию в той же сети. Это называется **Intra-LTE Handover**. Передача может также выполняться между провайдерами во время роуминга, называемая **Inter-LTE Handover**. Также возможна передача в другие сети (Inter-RAT Handover), например, перемещение между сотовым сигналом и сигналом Wi-Fi.

Если передача обслуживания происходит в одной и той же сети (передача обслуживания внутри LTE), то два eNodeB будут связываться через интерфейс X2, а основная сеть EPC не участвует в этом процессе. Если X2 недоступен, передача обслуживания должна управляться EPC через интерфейс S1. В каждом случае исходный eNodeB инициирует запрос транзакции. Если клиент выполняет Inter-LTE, передача обслуживания сложнее, так как задействованы два MME: источник (S-MME) и цель (T-MME).

Этот процесс позволяет осуществлять бесшовную передачу обслуживания, как показано в серии шагов на рис. 7.6. Во-первых, источник eNodeB примет решение о создании запроса на передачу обслуживания на основе емкости или перемещения клиента. eNodeB делает это, передавая сообщение MEASUREMENTCONTROLREQ в UE. Это отчеты об измерениях сети, отправленные при достижении определенных пороговых значений. Создается **Direct Tunnel Setup (DTS)**, где транспортный носитель X2 осуществляет связь между исходным eNodeB и целевым eNodeB. Если eNodeB определяет, что было бы целесообразно запустить передачу обслуживания, он находит целевой eNodeB адресата и отправляет RESOURCESTATUSREQUEST по интерфейсу X2, чтобы определить, может ли цель принять передачу обслуживания. Передча выполняется с помощью сообщения HANDOVERREQUEST. Целевой eNodeB готовит ресурсы для нового соединения. Исходный eNodeB отделит клиентское UE. Прямая пересылка пакетов из исходного eNodeB в eNodeB адресата гарантирует, что никакие пакеты в процессе не будут потеряны. Затем передача обслуживания завершается с помощью сообщения PATHSWITCHREQUEST между целевым eNodeB и MME, чтобы сообщить ему, что UE изменило соты. Носитель S1 будет освобожден, как и транспортный носитель X2, поскольку в это время в клиент EU не будут поступать пакеты вычетов.

Ряд шлюзовых устройств IoT, использующих 4G LTE, позволяет использовать несколько провайдеров (например, Verizon и ATT) на одном шлюзе или маршрутизаторе устройства. Такие шлюзы могут легко переключаться между передатчиками без потери данных. Это важная характеристика для мобильных и транспортных систем IoT, таких как логистика, аварийные транспортные средства и отслеживание объектов. Передача позволяет этим движущимся системам мигрировать между операторами для лучшего покрытия и эффективных тарифов.

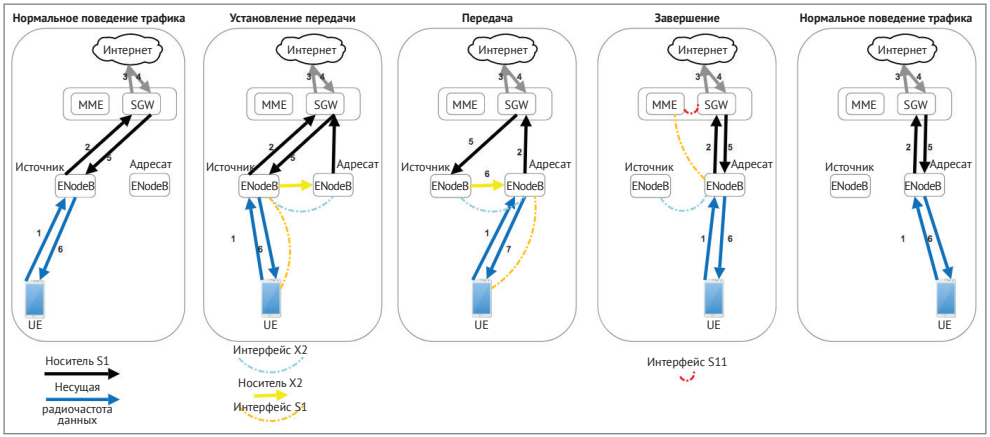


Рис. 7.6 ❖ Пример передачи обслуживания между двумя eNodeB

Структура пакета 4G LTE

Структура кадра пакета LTE похожа на другие модели OSI. На рис. 7.7 показано преобразование пакета с **PHY** до уровня **IP**. IP-пакет обернут слоями 4G LTE.

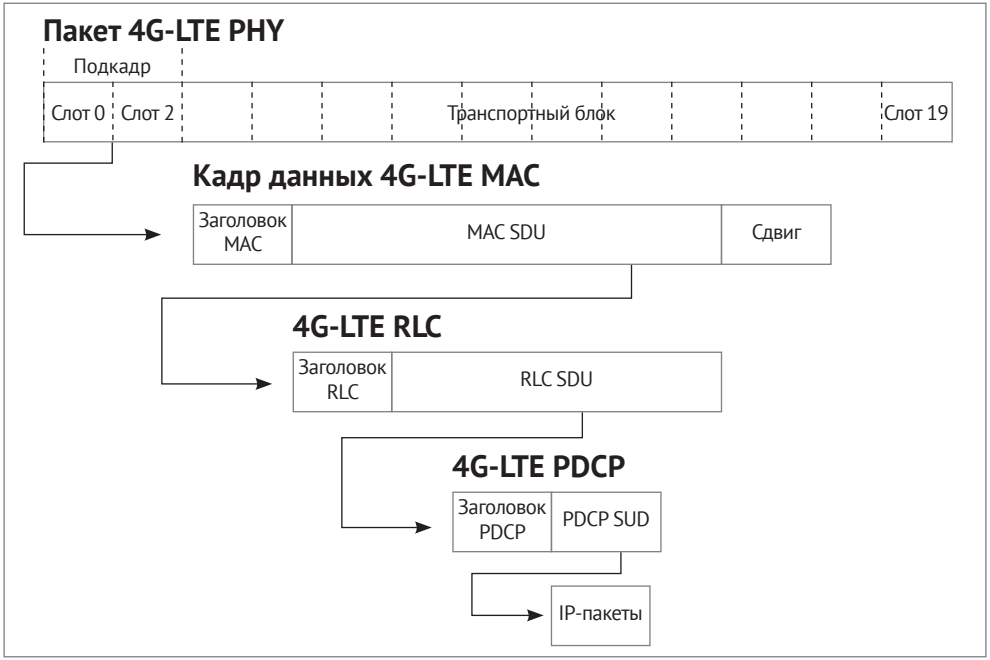


Рис. 7.7 ❖ Структура пакета 4G LTE. IP-пакеты преобразуются в SDUPDCP и проходят через уровни RLC, MAC и PHY. PHY создает слоты и подкадры транспортных блоков с уровня MAC

Категории 0, 1, M1 и NB-IoT

Соединение устройства IoT с интернетом отличается от типичных сотовых устройств на потребительском рынке, таких как смартфон. Смартфон в основном получает информацию из интернета по нисходящей линии связи. Часто это большие данные и потоковые данные в реальном времени, такие как видеоданные и музыка. При развертывании IoT данные могут быть очень разреженными и поступать короткими очередями. Чаще всего большинство данных будет генерироваться устройством и перемещаться по восходящей линии. Развитие LTE продвигалось в построении сотовой инфраструктуры и бизнес-модели, ориентированной и оптимизированной для мобильных потребителей. Новый шаг состоит в том, чтобы удовлетворить производителей данных IoT на краю, поскольку это увеличит количество потребителей. В следующих разделах рассматриваются низкомощные широкополосные сети (LPWAN) и, в частности, LTE, подобный LPWAN. Они подходят для развертывания IoT, но функции различаются.

До момента выхода релиза 13 наименьшая скорость передачи данных, подходящая для типичных устройств IoT, была Cat-1. Поскольку мобильная революция требовала более высоких скоростей и услуг, Cat-1 был пересмотрен в 3G и 4G. В релизах 12 и 13 были рассмотрены требования к устройству IoT для недорогих, маломощных передатчиков, разреженной передачи и расширений диапазона.

Общим для всех этих протоколов является то, что они все совместимы с существующей сотовой аппаратной инфраструктурой. Однако для включения новых функций для инфраструктуры необходимы изменения программного обеспечения в части стека протоколов. Без таких изменений UE Cat-0, Cat-M1 и Cat-NB даже не видят сеть. Архитектор IoT должен убедиться, что инфраструктура сотовой связи, которую он намеревается внедрить, была обновлена для поддержки этих стандартов.

Cat-1 не получил значительного распространения на рынке, и мы не будем вдаваться в спецификацию здесь, поскольку это похоже на материал 4G LTE, обсуждавшийся ранее.

LTE Cat-0

Cat-0 был представлен в релизе 12 и был первой архитектурой вне Cat-1, ориентированной на потребности IoT. Дизайн, как и многие другие спецификации LTE, основан на IP и работает в лицензированном спектре. Существенное различие заключается в пиковых скоростях передачи по восходящей линии и нисходящей линии связи (по 1 Мбит/с), в отличие от Cat-1 со скоростью 10 Мбит/с для нисходящей линии связи и 5 Мбит/с для восходящей линии связи. Хотя полоса пропускания канала остается 20 МГц, снижение скорости передачи данных значительно упрощает проектирование и снижает стоимость. Кроме того, переход от полного дуплекса к полудуплексной архитектуре дополнительно улучшает стоимость и мощность.

Обычно уровень NAS стека LTE не играет большой роли в обслуживании UE. В Cat-0 архитекторы 3GPP изменили возможности NAS, чтобы помочь в энергосбережении на уровне UE. Cat-0 вводит режим энергосбережения (**PSM**) в спецификацию LTE для решений с жесткими ограничениями по мощности. В традиционном устройстве LTE-модем остается подключенным к сотовой сети, потребляя энергию, вне зависимости от того, активно ли устройство, бездействует или спит. Во избежание превышения энергопотребления устройство может отключиться от сети и дизассоциироваться с сетью, но это может привести к повторному подключению и поиску в течение 15–30 с. PSM позволяет модему войти в очень глубокое состояние сна, когда он не активно общается, но может быстро при этом просыпаться. Он делает это, периодически выполняя обновление зоны отслеживания (**TAU**) и оставаясь доступным через пейджинг в течение программируемого периода времени. По сути, устройство IoT может входить в период бездействия на 24 часа и просыпаться один раз в день, чтобы передавать данные с датчика, все время оставаясь на связи. Все переговоры по настройке этих отдельных таймеров управляются с помощью изменений на уровне NAS и относительно просты в установке, если устанавливаются два таймера:

- **T3324** – время, которое UE остается в режиме ожидания. Если приложение устройства IoT не уверено, что есть ожидающие сообщения, оно может уменьшить значение таймера;
- **T3412** – как только таймер T3324 истечет, устройство войдет в PSM на время T3412. Устройство будет находиться в наименьшем возможном состоянии энергопотребления. Оно не может участвовать в пейджинговой или сетевой сигнализации. Устройство, однако, поддерживает все состояние UE (несущие, идентификаторы). Максимальное время – 12,1 дней.

При использовании PSM или других передовых режимов управления питанием целесообразно проверить соответствие инфраструктуры оператора. Некоторым системам сотовой связи требуется пинг каждые 2-4 часа для UE. Если оператор теряет связь в течение более 2-4 часов, он может считать UE недоступным и отсоединенным.

Cat-0 привлек мало внимания и мало распространен, его рост был ограниченным. Большинство новых свойств Cat-0 были включены в Cat-1 и другие протоколы.

LTE Cat-1

LTE Cat-1 повторно использует те же чипсеты и инфраструктуру несущих, что и Cat-4, поэтому его можно использовать по всей территории США с помощью инфраструктуры Verizon и AT&T. Cat-1 имеет значительную рыночную притягательность в отрасли M2M. Спецификация была частью восьмого релиза и позднее была обновлена для поддержки режима энергосбережения и единственной LTE-антенны Cat-0.

Cat-1 считается стандартом LTE со средней скоростью, что означает, что нисходящая линия связи составляет 10 Мбит/с, а восходящая линия – 5 Мбит/с.

При этих скоростях он по-прежнему способен передавать потоки голоса и видео, а также данные M2M и IoT. Приняв разработку Cat-0 PSM и антенны, Cat-1 работает с меньшей мощностью, чем традиционный 4G LTE. Также стало значительно дешевле разрабатывать радиостанции и электронику.

Cat-1 в настоящее время является лучшим выбором в сотовой связи для устройств IoT и M2M с самым широким охватом и самой низким энергопотреблением. Хотя он значительно медленнее, чем обычный 4G LTE (10 Мбит/с по сравнению с 300 Мбит/с по нисходящей линии связи), радио может быть спроектировано для возврата к 2G- и 3G-сети по мере необходимости. Cat-1 уменьшает сложность конструкции за счет включения разделения по времени, что также значительно снижает скорость.

Cat-1 можно рассматривать как дополнение к более новым узкополосным протоколам, которые рассматриваются далее.

LTE Cat-M1 (eMTC)

Cat-M1, также известный как усовершенствованные коммуникации машинного типа (а иногда и просто называемый Cat-M), был разработан для использования в случаях работы IoT и M2M с недорогими, маломощными расширениями с улучшенными диапазонами. Cat-M1 был выпущен в релизе 3GPP версии 13. Дизайн представляет собой оптимизированную версию архитектуры Cat-0. Единственное различие заключается в том, что ширина полосы канала уменьшается с 20 МГц до 1,4 МГц. Сокращение полосы пропускания канала с точки зрения оборудования ослабляет временные ограничения, мощность и схему. Затраты также снижаются на 33% по сравнению с Cat-0, поскольку схемам не требуется управлять широким спектром в 20 МГц. Другим значительным изменением является мощность передачи, которая уменьшается с 23 дБ до 20 дБ. Уменьшение мощности передачи на 50% снижает затраты, устраняя необходимость внешнего усилителя мощности и позволяя создавать одночиповую конструкцию. Даже при уменьшении мощности передачи покрытие улучшается на +20 дБ.

Cat-M1 наследует другие поздние протоколы 3GPP, основанные на IP-адресах. Хотя это не архитектура MIMO, пропускная способность может составлять 375 Кбит/с или 1 Мбит/с как по восходящей линии, так и по нисходящей линии связи. Архитектура обеспечивает мобильность и может применяться для связи с автомобилем или V2V. Ширина полосы пропускания достаточно велика, чтобы обеспечить голосовую связь, используя VoLTE. В сети Cat-M1 допускается применение нескольких устройств с использованием традиционного алгоритма SC-FDMA. Cat-M1 также использует более сложные функции, такие как скачкообразная перестройка частоты и турбо-кодирование.

Мощность имеет решающее значение для периферийных устройств IoT. Самым значительным фактором снижения мощности Cat-M1 является изменение мощности передачи. Как уже упоминалось, организация 3GPP снизила мощность передачи с 23 дБ до 20 дБ. Это снижение мощности не обязательно означает уменьшение диапазона. Сотовые башни будут ретранслировать паке-

ты от шести до восьми раз. Это делается для обеспечения приема в особо проблемных областях. Радиостанции Cat-M1 могут отключать прием, как только они получают пакет без ошибок.

Другой функцией энергосбережения является режим расширенного прерывистого приема (**eDRX**), который позволяет использовать период ожидания 10,24 с между циклами поискового вызова. Это значительно снижает мощность и позволяет UE спать в течение программируемого количества гиперкадров (**HF**) по 10,24 с каждый. Устройство может войти в этот расширенный режим ожидания в течение до 40 минут. Это позволяет радио использовать ток холостого хода до 15 мкА.

Дополнительные возможности по снижению мощности включают:

- PSM, как представлено в Cat-0 и релизе 13;
- измерения отдыхающих соседних ячеек и периоды отчетности. Если устройство IoT находится в стационарном состоянии или медленном движении (датчик в здании, крупный рогатый скот в кормовом канале), тогда инфраструктура вызова может быть настроена для ограничения управляющих сообщений;
- функция контрольной панели пользователя и управления CIo TEPS является частью RRC в стеке E-UTRAN. В обычных LTE-системах новый RRC-контекст должен создаваться каждый раз, когда UE просыпается из режима IDLE. Это потребляет большую часть мощности, когда устройству просто нужно отправить ограниченный объем данных. Используя оптимизацию EPS, контекст RRC сохраняется;
- сжатие заголовков пакетов TCP или UDP;
- сокращение времени синхронизации после длительных периодов сна.

Следующий раздел посвящен Cat-NB. На рынке сложилось мнение, что Cat-NB значительно ниже по мощности и стоимости по сравнению с другими протоколами, такими как Cat-M1. Это отчасти верно, и архитектор IoT должен понимать варианты использования и тщательно выбирать, какой протокол использовать. Например, если мы рассматриваем постоянную мощности передачи между Cat-NB и Cat-M1, мы видим, что Cat-M1 имеет коэффициент усиления в 8 дБ. Другим примером является мощность; в то время как Cat-M1 и Cat-NB имеют аналогичные и прогрессивные функции управления питанием, Cat-M1 будет использовать меньше энергии для больших размеров данных. Cat-M1 может передавать данные быстрее, чем Cat-NB, и быстрее входить в состояние глубокого сна. Это та же самая концепция Bluetooth 5, которая используется для снижения мощности, просто посылая данные быстрее, а затем переходя в состояние ожидания. Кроме того, на момент написания статьи Cat-M1 доступен сегодня, а Cat-NB не соответствует рынкам США.

LTE Cat-NB

Cat-NB, также известный как NB-IoT, NB1 или IoT с узким диапазоном, является другим протоколом LPWAN, управляемым 3GPP в релизе 13. Как и Cat-M1, Cat-NB работает в лицензированном спектре. Как и Cat-M1, целью является со-

кращение мощности (10-летнее время работы батареи), расширение покрытия (+20 дБ) и снижение стоимости (5 долларов США за модуль). Cat-NB основан на **Evolved Packet System (EPS)** и оптимизации для **Cellular Internet of Things (CIoT)**. Поскольку каналы намного более узкие, чем даже 1,4-мегагерцовый Cat-M1, стоимость и мощность могут быть заметно упрощены благодаря более простым конструкциям аналого-цифровых и цифро-аналоговых преобразователей.

Значительные различия между Cat-NB и Cat-M1 включают следующие различия:

- **очень узкая полоса пропускания канала** – как и в случае Cat-M1, в котором ширина канала уменьшена до 1,4 МГц, Cat-NB уменьшает его до 180 кГц по тем же причинам (снижая затраты и мощность);
- **нет VoLTE** – поскольку канал настолько узок, у него нет возможности для голосового или видео трафика;
- **нет мобильности** – Cat-NB не поддерживает передачу обслуживания и должен оставаться связанным с одной ячейкой или оставаться неподвижным. Это отлично подходит для большинства защищенных и закрепленных датчиков IoT. Это касается передачи обслуживания другим ячейкам и другим сетям.

Независимо от этих существенных различий Cat-NB основан на мультиплексировании OFDMA (нисходящей линии связи) и SC-FDMA (восходящей линии связи) и использует один и тот же интервал поднесущей и длительность символа. Стек протокола E-UTRAN также совпадает с типичными уровнями RLC, RRC и MAC и остается основанным на IP, но считается новым воздушным интерфейсом для LTE.

Так как ширина канала настолько мала (180 кГц), это дает возможность заключить сигнал Cat-NB внутри большего LTE-канала, заменить канал GSM или даже существовать в защитном канале обычных сигналов LTE. Это обеспечивает гибкость в развертывании LTE, WCDMA и GSM. Опция GSM является самой простой и быстрой на рынке. Некоторые части существующего трафика GSM могут быть размещены в сети WCDMA или LTE. Это освобождает провайдера GSM для трафика IoT. In-band обеспечивает огромный спектр для использования, поскольку полосы LTE намного больше, чем полоса 180 кГц. Это позволяет на практике развертывать до 200 000 устройств на каждой ячейке. В этой конфигурации базовая сотовая станция будет мультиплексировать данные LTE с трафиком Cat-NB. Это вполне возможно, так как архитектура Cat-NB является автономной сетью и прекрасно взаимодействует с существующей инфраструктурой LTE. Наконец, использование Cat-NB в качестве защитной полосы LTE является уникальной и новой концепцией. Поскольку архитектура повторно использует одну и ту же конструкцию поднесущих 15 кГц, ее можно обеспечить с помощью существующей инфраструктуры.

На рис. 7.8 показано, где разрешено располагаться сигналу.

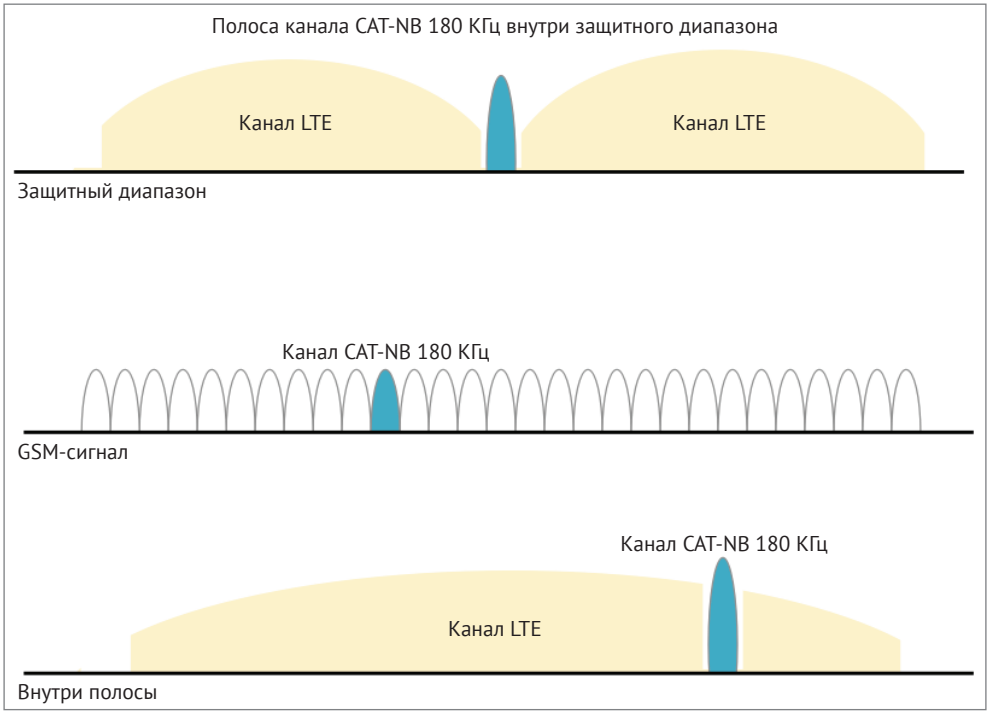


Рис. 7.8 ❖ Варианты развертывания Cat-NB в качестве защитного диапазона, внутри GSM-сигнала или внутри полосы с LTE

Так как **максимальная потеря сцепления (MCL)** составляет 164 дБ, она обеспечивает глубокое покрытие в подвалах, туннелях, сельских районах и открытых средах. Повышение на 20 дБ по сравнению со стандартным LTE – это увеличение площади покрытия в 7 раз. Доступная скорость передачи данных является функцией SNR и ширины канала, как мы видели в теореме Шеннона-Хартли. Для связи по восходящей линии связи Cat-NB назначит каждому UE одну или более поднесущих 15 кГц в блоке ресурсов 180 кГц. Cat-NB имеет возможность уменьшить ширину поднесущей до 3,75 кГц, что позволяет другим устройствам разделять пространство. Однако необходимо тщательно проверить потенциал помех между поднесущими между 3,75 кГц и следующей поднесущей 15 кГц.

Скорость передачи данных, как мы узнали, является функцией покрытия. Эриксон провел тесты, иллюстрирующие влияние различного покрытия и силы сигнала. Данные раскрывают и объясняют, почему латентность может стать важной проблемой для Cat-NB.

На границе ячейки: Coverage = +0 dB, Uplink Time = 39 мс,
Общее время передачи = 1,604 мс.
При среднем охвате: охват = +10 дБ, время восходящей линии = 553 мс, общее время передачи = 3,085 мс.

Наихудший случай: покрытие: +20 дБ, время восходящей линии = 1,923 мс, общее время передачи = 7623 мс.
Источник: «NB-IOT: устойчивая технология для соединения миллиардов устройств» // Ericcson Technology Review. Vol. 93. Стокгольм, Швеция. № 3. 2016.

Управление питанием очень похоже на Cat-M1. Все технологии управления питанием в версиях 12 и 13 применяются также к Cat-NB (PSM, eDRX и все остальные функции).

5G

5G (или 5G-NR для нового радио) является стандартом для IP-коммуникаций нового поколения, разработанным и предназначенным для замены 4G LTE. Тем не менее, он использует некоторые технологии 4G LTE, но имеет существенные отличия и новые возможности. По 5G написано так много материала, что он затмевает даже текущий материал по Cat-1 и Cat-M1. В частности, 5G обещает предоставить существенные возможности для IoT, коммерческих, мобильных и транспортных применений. 5G также улучшает пропускную способность, латентность, плотность и стоимость для пользователя. Вместо того, чтобы создавать разные сотовые услуги и категории для каждого варианта использования, 5G пытается стать одним зонтичным стандартом, чтобы обслуживать их всех. Между тем, 4G LTE будет оставаться доминирующей технологией для покрытия сотовой связи и будет продолжать развиваться. 5G не является продолжением эволюции 4G; он происходит от 4G, но представляет собой новый набор технологий. В этом разделе мы рассмотрим только те элементы, которые относятся к случаям использования IoT или заслуживают внимания, и могут стать частью спецификации 5G.

5G стремится к первому подключению потребителя в 2020 г., однако массовое развертывание и переход могут последовать спустя годы в середине 2020-х гг. Цели и архитектура 5G все еще развиваются, они начались с 2012 г. Для 5G было три различные и разные цели: www.gsmhistory.com/5g/. Цели варьируются от конвергентной волоконно-оптической и сотовой инфраструктуры, сверхбыстрых мобильных телефонов с использованием небольших ячеек и сниженных барьеров стоимости мобильных устройств, что является частью истории 5G. Также ITU-R одобрил международные спецификации и стандарты, в то время как 3GPP отвечает стандарту, соответствующему временной шкале ITU-R. 3GPP RAN уже начал анализировать учебные статьи по релизу 14. Целью является создание двухфазного выпуска технологий 5G. Первый этап будет завершен в 2018 г. с релизом 15, а второй этап запланирован на релиз 19 в 2019 г. Оба они совпадают с целью коммерческого запуска в 2020 г.

Verizon начала коммерческое развертывание 5G в 2018 г. (спецификация перед 3GPP). В 3GPP есть ускоренный 5G как не автономный (**NSA**) и автономный (**SA**). NSA будет использовать ядро LTE, а SA будет использовать Core 5G следующего поколения. 5G для конкретных случаев использования IoT – это вопрос нескольких лет, поскольку первоначальный фокус на рынке – мобильные устройства и смартфоны.

Общий консенсус особенностей в 5G, которые следует понимать и которые имеют хорошую перспективу быть в окончательном релизе (рис. 7.9):

- расширенная мобильная широкополосная связь (eMBB):
  - от 1 до 10 Гбит/с соединения с UE/оконечными точками в поле (не теоретически);
  - 100%-ный охват по всему миру (или восприятие);
  - от 10 до 100 – количество подключенных устройств через 4G LTE;
  - возможность подключения на скорости 500 км/ч;
- ультранадежная и связь с низкими задержками (URLLC):
  - Sub < 1 мс сквозная латентность в оба конца;
  - доступность 99,999% (или восприятие);
- массивные сообщения машинного типа (mMTC):
  - 1000x полоса пропускания на единицу площади; это подразумевает примерно 1 миллион узлов в 1 км^2;
  - до 10 лет работы батарей на оконечных точках IoT;
  - снижение энергопотребления сети на 90%.

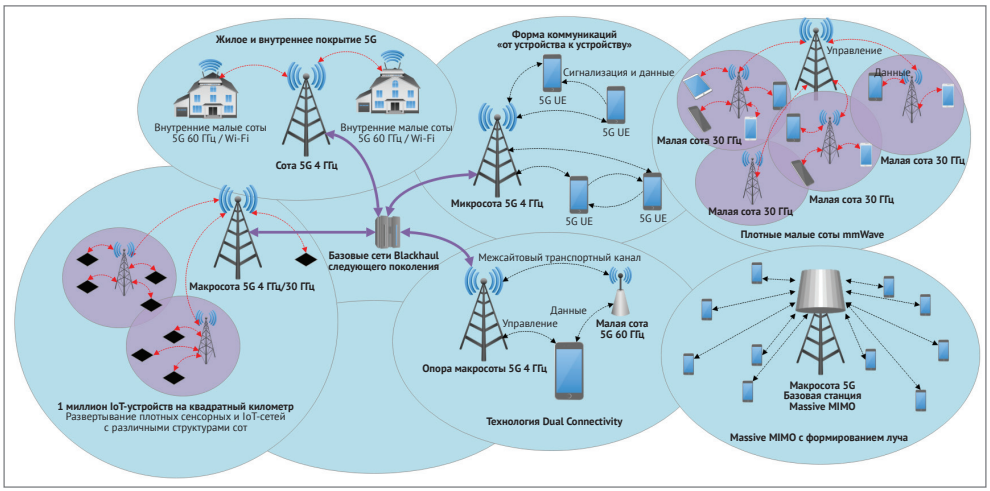


Рис. 7.9 ❖ 5G-топологии. Слева направо: плотность узлов 1 миллион через небольшие ячейки и развертывание макроячейки. Внутреннее и домашнее использование 60 ГГц с макроячейкой при 4 ГГц. Пример с двумя соединениями с разделенным управлением и плоскостями данных с использованием двух радиостанций для пользовательских данных и макроячеек 4 ГГц для плоскости управления. Подключение устройства к устройству. Массивный MIMI с формированием луча от одной mmWave антенны. Увеличение плотности с помощью комбинации небольших ячеек в mmWave для полного покрытия пользовательских данных

В современных системах 4G LTE используются частоты ниже диапазона 3 ГГц. 5G радикально изменит использование спектра. В то время как про-

странство ниже 3 ГГц значительно перегружено и нарезано участками полосы пропускания, 5G может использовать множество частот. При пристальном рассмотрении используются **миллиметровые волны (mmWave)** в нелицензированном диапазоне от 24 до 100 ГГц. Эти частоты непосредственно касаются закона Шеннона, увеличивая полосу пропускания B закона с чрезвычайно широкими каналами. Поскольку пространство mmWave не насыщено или не нарезано различными регулирующими органами, возможны каналы шириной до 100 МГц на частотах от 30 до 60 ГГц. Это обеспечит технологию для поддержки скорости передачи данных в несколько гигабит в секунду.

Основные проблемы с технологией mmWave – потеря свободного пространства, затухание и проникновение. Если мы помним, что потеря свободного пробега может быть рассчитана как $L_{fs} = 32{,}4 + 20\log_{10}f + 20\log_{10}R$ (где f – частота, а R – диапазон), то мы можем видеть, как на потери влияют 2,4, 30 и 60 ГГц:

- 2,4 ГГц, в пределах 100 м: 80,1 дБ;
- 30 ГГц, в пределах 100 м: 102,0 дБ;
- 60 ГГц, в пределах 100 м: 108,0 дБ;
- 2,4 ГГц, в пределах 1км: 100,1 дБ;
- 30 ГГц, в пределах 1 км: 122,0 дБ;
- 60 ГГц, в пределах 1 км: 128,0 дБ.

20 дБ значительны, но с mmWave антенны могут размещать значительно больше антенных элементов, чем антенна с частотой 2,4 ГГц. Свободные потери на трассе значительны только в том случае, если коэффициент усиления антенны не зависит от частоты. Если мы сохраним площадь антенны постоянной, можно уменьшить влияние потерь на трассе. Для этого требуется технология **Massive-MIMO (M-MIMO)**. M-MIMO будет включать в себя башни с макроячейками с антеннами от 256 до 1024. Также будет использоваться формирование луча на макроячейке. В сочетании с mmWaves M-MIMO имеет проблемы с влиянием на соседние башни, а мультиплексирующие протоколы, такие как TDD, должны быть перепроектированы.

Еще одна проблема с 5G – необходимость в очень больших антенных решетках для поддержки M-MIMO с сотнями антенн в плотной конфигурации башни. Рассматриваются плотно упакованные трехмерные структуры антенн для поддержки формирования луча на башне. Проблемы, такие как воздействие ветра и шторма на эти башни, по-прежнему необходимо решить.

Ослабление – очень важная проблема. При 60 ГГц сигнал будет поглощаться кислородом в атмосфере. Даже вегетация и само человеческое тело окажут серьезное влияние на сигналы. Человеческое тело поглотит так много радиочастотной энергии на 60 ГГц, что оно создаст тень. Сигнал на частоте 60 ГГц будет иметь потерю 15 дБ/км. Таким образом, связь на дальнем расстоянии будет страдать при использовании 5G на частоте 60 ГГц и потребует либо покрытия из небольших ячеек, либо переход на более медленное частотное пространство. Это одна из причин того, что для архитектуры 5G потребуются несколько диапазонов, небольшие ячейки, макроячейки и гетерогенная сеть.

Наконец, проницаемость материала в спектре mmWave является сложной задачей. mmWave-сигналы ослабляются через гипсокартон на 15 дБ, а стеклянные окна в зданиях способствуют снижению на 40 дБ. Поэтому закрытое покрытие с макроячейкой практически невозможно. Эти и другие типы проблем сигнализации будут смягчены благодаря широкомасштабному использованию внутренних небольших ячеек (рис. 7.10).

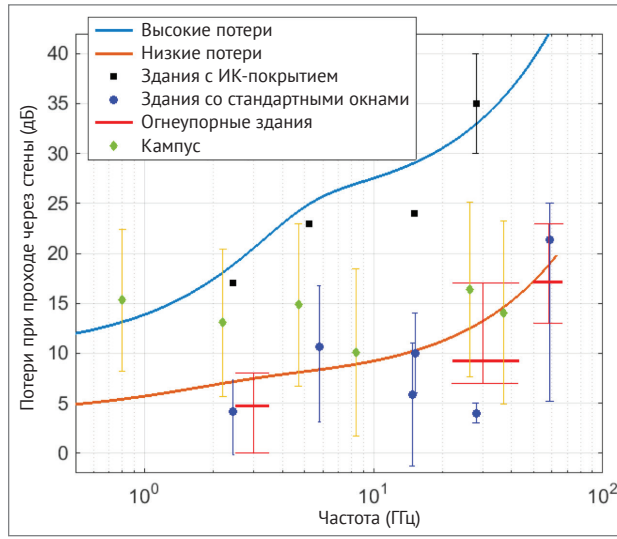


Рис. 7.10 ❖ Различные частоты и потери при пропуске (дБ). Типовые строительные композиционные материалы были протестированы (стекло, кирпич, дерево) из внешнего во внутреннее пространство. Потеря восстановительного стекла в инфракрасном диапазоне особенно затруднительна для частот mmWave. Источник: Aalto University и др., «Модель канала 5G для полос до 100 ГГц», 3-й семинар по мобильной связи в высокочастотных диапазонах (MCHFB), «Заметки», декабрь, 2016 г.

UE могут использовать несколько диапазонов одновременно. Например, оконечное устройство может использовать более низкие частоты для связи на большие расстояния и переключиться на mmWave для внутренней и личной связи. Другая рассматриваемая схема – *Dual Connectivity*. Dual Connectivity управляет трафиком данных на нескольких диапазонах в зависимости от типа данных. Например, плоскость управления и пользовательская плоскость стека уже разделены. Данные пользовательской плоскости могут направляться в ближайшую небольшую ячейку с использованием частоты 30 ГГц, в то время как данные управления могут быть направлены на многоцелевую башню eNodeB с обычной частотой 4 ГГц.

Другим улучшением для увеличения скорости является спектральная эффективность. 3GPP фокусируется на определенных правилах проектирования:

- интервал 15 кГц для повышения мультиплексирования;
- гибкая и масштабируемая нумерология символов от 2^M символов до 1 символа для уменьшения латентности.

Как обсуждалось ранее, спектральная эффективность дана как бит/с/Гц, используя D2D и M-MIMO для улучшения спектральной эффективности наряду с изменениями в радиоинтерфейсе и новом радио. 4G LTE использует OFDM, который хорошо работает для больших передач данных. Однако для IoT и mMTC пакеты намного меньше. Накладные расходы для OFDM также влияют на латентность в очень плотных развертываниях IoT. Следовательно, новые формы волн создаются для рассмотрения:

- **неортогональный множественный доступ (NOMA)** – позволяет нескольким пользователям совместно использовать беспроводную среду;
- **Filter Bank Multi-Carrier (FBMC)** – управляет формой сигналов M поднесущих для удаления боковых лепестков через DSP;
- **разрешенный множественный доступ с ограниченным кодированием (SCMA)** – позволяет сопоставлять данные с другим кодом из разных кодовых книг.

Снижение латентности также является целью ITU и 3GPP. Снижение латентности имеет решающее значение для использования 5G в таких случаях, как интерактивные развлечения и гарнитуры виртуальной реальности, а также для промышленной автоматизации. Однако это играет большую роль в снижении энергопотребления (другая цель ITU). 4G LTE может иметь задержку до 15 мс на подкадрах 1 мс. 5G готовится к задержке суб-1 мс. Это также будет выполнено с использованием небольших ячеек для маршрутизации, а не для перегруженных макроячеек. Архитектура также планирует связь между **устройством и устройством (D2D)**, по существу, убирая инфраструктуру ячеек из пути передачи данных для связи между UE.

Системы 4G будут продолжать существовать, так как развертывание 5G займет несколько лет. Будет существовать период сосуществования, которое необходимо установить. В релизе 15 будут добавлены дополнительные определения для общей архитектуры, такие как выбор каналов и частот. С точки зрения архитектора IoT, 5G – это технология наблюдения и планирования. Устройства IoT могут предсказать глобальную сеть, которая должна работать в течение десятка лет или более в полевых условиях. Хорошая точка зрения на ключевые моменты, ограничения и подробный дизайн 5G, выражена в следующем документе: *A Tutoria lOverview of Standards, Trials, Challenges, Deployment and Practice, M. Shafi и др., в IEEE Journal по отдельным областям в области связи, том 35, вып. 6, с. 1201–1221, июнь, 2017 г.*

LoRa и LoRaWAN

LPWAN также включает технологии, которые являются проприетарными и неспонсируемыми 3GPP. Можно утверждать, что некоторые протоколы IEEE 802.11 также следует классифицировать в LPWAN, но в следующих двух раз-

делах будут показаны LoRa и Sigfox. LoRa – это физический уровень для протокола IoT на большие расстояния и малой мощности, в то время как LoRaWAN представляет уровень MAC.

Эти проприетарные технологии и носители LPWAN имеют преимущество использования нелицензионного спектра, и это просто стоимость данных. Как правило, технологии, такие как Sigfox и LoRaWAN, будут в 5–10 раз ниже в скорости передачи данных по сравнению с традиционными соединениями 3G или LTE для развертывания большого объема (>100 000 единиц). Это может измениться с большей конкуренцией со стороны Cat-M1, Cat-NB и Cat-5, но пока еще слишком рано говорить об этом.

Архитектура была первоначально разработана Cycleo во Франции, но затем приобретена Semtech Corporation (французским производителем электроники смешанных сигналов) в 2012 г. за 5 миллионов долларов наличными. Альянс LoRa был сформирован в марте 2015 г. Альянс является стандартизующим органом для спецификации и технологии LoRaWAN. Туда также входит процесс соблюдения и сертификации для обеспечения совместимости и соответствия стандарту. Альянс поддерживается IBM, Cisco и более чем 160 другими участниками.

LoRaWAN заработала в Европе с развертыванием сетей KPN, Proximus, Orange, Bouygues, Senet, Tata и Swisscom. В других областях пока что имеется слабое влияние.

Одной из причин разницы в цене, помимо того, что она находится в нелицензированном спектре, является то, что один шлюз LoRaWAN может покрыть значительную площадь. Бельгия площадью 30 500 км^2 полностью покрыта семью шлюзами LoRaWAN. Типичный диапазон от 2 до 5 км в городских районах и 15 км в пригородных районах. Это снижение стоимости инфраструктуры сильно отличается от 4G LTE с гораздо меньшими ячейками.

Поскольку LoRa является нижней частью стека, он был принят в конкурирующих архитектурах в LoRaWAN. Например, SymphonyLink – это решение LPWAN от Link Labs на основе LoRaPHY, использующее восьмиканальную базовую станцию с субгигагерцами для промышленных и муниципальных развертываний IoT. Еще одним конкурентом, использующим LoRa, является Haystack, который производит систему DASH7. DASH7 – полный сетевой стек на LoRaPHY (а не только уровень MAC).

Следующий раздел будет посвящен исключительно LoRaWAN.

Физический уровень LoRa

LoRa представляет физический уровень сети LoRaWAN. Он управляет модуляцией, мощностью, приемником и передающими радиостанциями, а также формирует сигналы.

Архитектура основана на следующих диапазонах в пространстве ISM без лицензирования:

- **915 МГц** – в США с ограничениями мощности, но без ограничения рабочего цикла;

- **868 МГц** – в Европе с 1%-ным и 10%-ным рабочим циклом;
- **433 МГц** – в Азии.

Производным от **Chirp Spread Spectrum (CSS)** является метод модуляции, используемый в LoRa. CSS балансирует скорость передачи данных с чувствительностью в полосе фиксированного канала. CSS был впервые использован в 1940-х гг. для военной длинноволновой связи с использованием модулированных импульсов *чирпа* для кодирования данных и был признан особенно устойчивым к помехам, эффектам Доплера и многолучевому распространению. Чирпы – это синусоидальные волны, которые со временем увеличиваются или уменьшаются. Поскольку они используют весь канал для связи, они относительно надежны в плане помех. Мы можем представить сигналы чирпов с увеличением или уменьшением частот (звук, как зов кита). Частота передачи битов – битрейт, где LoRa является функцией скорости чирпа и скорости передачи символов. Битрейт представлен R, коэффициент расширения S, полоса пропускания B. Поэтому битрейт (бит/с) может варьироваться от 0,3 до 5 Кбит/с и выводится как:

$$R_b = S \times \frac{1}{\left[\frac{2^S}{B}\right]}.$$

Такая форма модуляции допускает малую мощность для больших расстояний, как показали военные. Данные кодируются с использованием увеличения или уменьшения частоты, и несколько передач могут быть отправлены с разной скоростью передачи данных на той же частоте. CSS позволяет получать сигналы на уровне 19,4 дБ ниже уровня шума, используя FEC. Группа также подразделяется на несколько поддиапазонов. LoRa использует каналы 125 кГц и выделяет шесть каналов 125 кГц и скачкообразную пересылку псевдослучайных каналов. Кадр будет передаваться с определенным коэффициентом расширения. Чем выше коэффициент расширения, тем медленнее передача, но тем длиннее диапазон передачи. Кадры в LoRa являются ортогональными, что означает, что несколько кадров могут отправляться одновременно, пока каждый отправляется с другим коэффициентом расширения. Всего имеется шесть различных коэффициентов расширения (от SF = 7 до SF = 12).

Типичный пакет LoRa содержит преамбулу, заголовок и полезную нагрузку от 51 до 222 байт.

Сети LoRa имеют мощную функцию, называемую **Adaptive Data Rate (ADR)**. По сути, это позволяет динамически масштабировать емкость, основываясь на плотности узлов и инфраструктуре. ADR контролируется управлением сетью в облаке. Узлы, близкие к базовой станции, могут иметь более высокую скорость передачи данных из-за достоверности сигнала. Узлы, находящиеся в непосредственной близости, могут передать данные и освободить свою полосу пропускания и быстро войти в состояние сна по сравнению с удаленными узлами, которые передают с меньшей скоростью.

В табл. 7.5 описаны свойства восходящей и нисходящей линии связи.

Таблица 7.5. Свойства восходящей и нисходящей линии связи

Свойство	Восходящее соединение	Нисходящее соединение
Модуляция	CSS	CSS
Бюджет связи	156 дБ	164 дБ
Скорость передачи (адаптивная)	От 0,3 до 5 кб/с	От 0,3 до 5 кб/с
Размер сообщения на полезную нагрузку	0–250 байт	0–250 байт
Длительность сообщения	От 40 мс до 1,2 с	От 20 до 160 мс
Энергия, затраченная на сообщение	E_{tx} = 1,2s * 32 мА = 11uAh при полной чувствительности приема E_{tx} = 40 мс * 32 мА = 0,36uAh при минимальной чувствительности приема	E_{tx} =160 мс * 11 mA = 0,5 uAh

Уровень MAC LoRaWAN

LoRaWAN представляет MAC, который находится поверх LoRaPHY. MAC-адрес LoRaWAN является открытым протоколом, в то время как PHY закрыт. Существует три протокола MAC, которые являются частью уровня канала передачи данных. Все три балансируют задержки и использование энергии. Класс A является наилучшим для уменьшения энергопотребления при максимальной задержке. Класс B находится между классом A и классом C. Класс C имеет минимальную задержку, но самый высокий уровень использования энергии.

Устройства класса A – это датчики и оконечные точки на аккумуляторах. Все оконечные точки, которые соединяются с сетью LoRaWAN, сначала ассоциируются как класс A с возможностью изменения класса во время работы. Класс A оптимизирует питание, устанавливая различные задержки приема во время передачи. Оконечная точка начинает с отправки пакета данных в шлюз. После передачи устройство переходит в состояние ожидания до истечения времени ожидания таймера приема. Когда время таймера истекает, оконечная точка просыпается и открывает слот приема и ожидает передачу в течение определенного периода времени, а затем снова переходит в состояние ожидания. Устройство снова проснется по истечении другого таймера. Это означает, что вся нисходящая связь происходит в течение короткого периода времени после того, как устройство отправляет пакет по восходящей линии. Этот период может быть очень долгим.

Устройства класса B балансируют мощность и задержку. Этот тип устройства полагается на сигнал маяка, отправляемый шлюзом через равные промежутки времени. Маяк синхронизирует все оконечные точки в сети и транслируется в сеть. Когда устройство получает сигнал маяка, оно создает слот для проверки, который является коротким окном приема. Во время этих периодов вы можете отправлять и получать сообщения. В любое другое время устройство находится в состоянии ожидания. По сути, это сеанс, инициированный шлюзом, и основанный на методе связи слотами.

Оконечные точки класса C используют наибольшую мощность, но имеют самую короткую задержку. Эти устройства открывают два окна приема класса A, а также окно приема с включенным питанием. Устройства класса C обычно включаются и могут быть исполнительными устройствами или подключаемыми устройствами. Для передачи по нисходящей линии нет задержки. Устройства класса C не могут реализовывать класс B.

Стек протокола LoRa/LoRaWAN можно визуализировать так, как показано в табл. 7.6.

Таблица 7.6. Стек протоколов LoRa и LoRaWAN. Сравнение со стандартной моделью OSI. Примечание: LoRa/LoRaWAN представляет только уровни 1 и 2 модели стека

<table>
<tr><th colspan="3">LoRa/LoRaWAN Protocol Stack</th><th>Simplified OSI Model</th></tr>
<tr><td colspan="3">Прикладной уровень</td><td>7. Прикладной уровень</td></tr>
<tr><td colspan="3">LoRaWAN Layer</td><td rowspan="2">2. Канальный Layer</td></tr>
<tr><td>Class-A (Baseline)</td><td>Class-B (Baseline)</td><td>Class-C (Continuous)</td></tr>
<tr><td colspan="3">Lora PHY Modulation</td><td rowspan="3">1. Физический Layer</td></tr>
<tr><td colspan="3">Lora PHY Regional ISM Band</td></tr>
<tr><td>Lora PHY
EU Band 868 MHz</td><td>Lora PHY
EU Band 433 MHz</td><td>Lora PHY
EU Band 915 MHz</td></tr>
</table>

Для безопасности LoRaWAN шифрует данные с использованием модели AES128. Одно из отличий в безопасности от других сетей – LoRaWAN отделяет аутентификацию и шифрование. Аутентификация использует один ключ (NwkSKey), а пользовательские данные – отдельный ключ (AppSKey). Чтобы подсоединиться к сети LoRa, устройства отправят запрос JOIN. Шлюз ответит адресом устройства и маркером аутентификации. Ключ приложения и сетевого сеанса будет получен во время процедуры JOIN. Этот процесс называется **Over The Air Activation (OTAA)**. В качестве альтернативы устройство на основе LoRa может использовать *активацию посредством персонализации*. В этом случае поставщик/оператор LoRaWAN предварительно распределяет 32-разрядные сетевые и сеансовые ключи, и клиент должен заказать план подключения и соответствующий набор ключей. Ключи будут заказываться у изготовителя оконечной точки с ключами, встроенными в устройство.

LoRaWAN – это асинхронный протокол на основе ALOHA. Чистый протокол ALOHA был первоначально разработан в Гавайском университете в 1968 г. как форма связи с множественным доступом до тех пор, пока не существовали такие технологии, как CSMA. В ALOHA клиенты могут передавать сообщения, не зная, находятся ли другие клиенты в процессе передачи одновременно. Нет никаких оговорок или методов мультиплексирования. Основным принципом является хаб (или шлюз в случае LoRaWAN), который немедленно ретранслирует полученные пакеты. Если оконечная точка замечает, что один из ее пакетов не был подтвержден, он будет ждать, а затем повторно передаст пакет.

В LoRaWAN коллизии возникают только в том случае, если при передачах используются одни и те же каналы и частота распространения.

Топология LoRaWAN

LoRaWAN основан на топологии звезды. В этом отношении можно сказать, что он поддерживает звезду топологии звезд. Модель LoRaWAN может использоваться не как одна модель с хабом, а как несколько хабов. Узел может быть связан с несколькими шлюзами.

Критическим компонентом LoRaWAN, который отличает его от большинства транспортов, перечисленных в этой книге, является тот факт, что данные пользователя будут переноситься с оконечного узла на шлюз по протоколу LoRaWAN. В этот момент шлюз LoRaWAN отправит пакет по любому транспортному объекту (например, 4G LTE, Ethernet, Wi-Fi) на выделенную сетевую услугу LoRaWAN в облаке. Это уникально, поскольку большинство других архитектур WAN не осуществляют какой-либо контроль над данными о клиентах в то время, когда они покидают свою сеть до места назначения в интернете.

Сетевая служба имеет правила и логику для обслуживания необходимых верхних уровней сетевого стека. Побочным эффектом этой архитектуры является то, что передача обслуживания от одного шлюза к другому не требуется. Если узел мобилен и перемещается от антенны к антенне, сетевые службы будут захватывать несколько идентичных пакетов с разных путей. Эти облачные сетевые службы позволяют системам LoRaWAN выбирать лучший маршрут и источник информации, когда оконечный узел связан с несколькими шлюзами. Обязанности сетевых служб включают:

- идентификацию и удаление повторного пакета;
- услуги безопасности;
- маршрутизацию;
- сообщения о подтверждении.

Кроме того, LPWAN-системы, такие как LoRaWAN, будут иметь на 5x–10x меньше базовых станций для аналогичного покрытия сети 4G. Все базовые станции прослушивают один и тот же набор частот, поэтому они логически являются одной очень большой базовой станцией. Это, конечно же, подтверждает утверждение о том, что системы LPWAN могут иметь более низкую стоимость, чем традиционная сотовая сеть (рис. 7.11).

Краткое описание LoRaWAN

У LoRaWAN есть пробелы в архитектуре и протоколах, которые требуют тщательного рассмотрения от архитектора IoT:

- некоторые функции, общие для сетей LTE, просто не предназначены для архитектуры LoRaWAN. LoRaWAN не является полным сетевым стеком в модели OSI. Он не имеет общих черт сетевого уровня, уровня сеанса и транспортного уровня: роуминга, пакетирования, механизмов повтора, QoS и разъединений. При необходимости разработчик и интегратор могут добавить эти услуги;

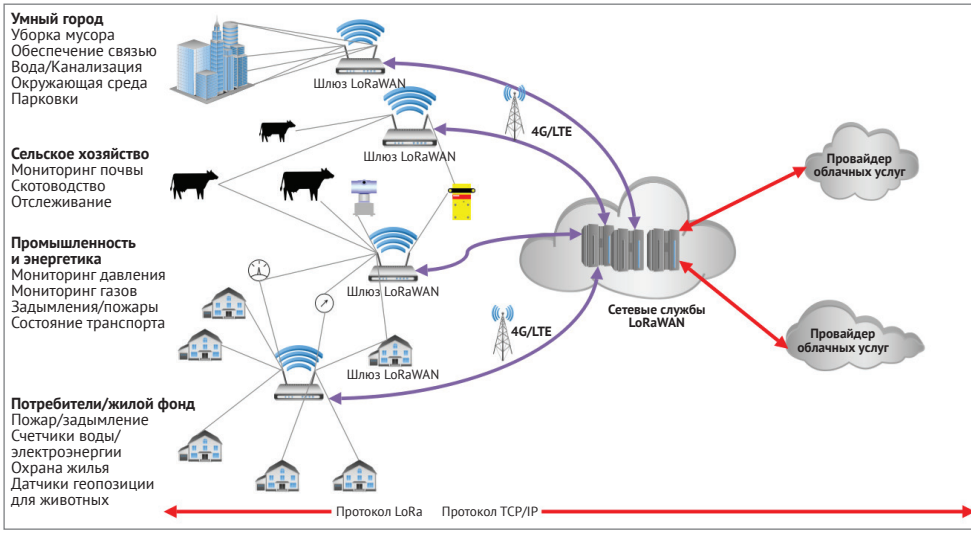


Рис. 7.11 ❖ Сетевая топология LoRaWAN.
LoRaWAN построен на топологии звезды со шлюзами, действующими в качестве концентраторов, а также агентами связи по сравнению с традиционными IP-сетями для администрирования LoRaWAN в облаке. Несколько узлов могут связываться с несколькими шлюзами LoRaWAN

- LoRaWAN опирается на облачный сетевой интерфейс. В какой-то момент в цепочке вычисления стоимости необходима облачная подписка;
- поставщик чипов Semtech и является единственным источником технологии, хотя были объявлены партнерские отношения с STMicroelectronics. Это имеет схожесть cZ-Wave;
- LoRaWAN основан на протоколе ALOHA. ALOHA усложняет проверку и подтверждение, а также способствует повышению частоты ошибок более чем на 50%;
- возможности по нисходящей линии связи по-прежнему ограничены. LoRaWAn является принципиально односторонним протоколом. В некоторых случаях для использования достаточно, но это требует гибкости;
- LoRaWAN обладает высокой задержкой и не может работать в режиме реального времени;
- нет обновлений прошивки OTA;
- мобильность и движущиеся узлы сложнее управлять в LoRaWAN. Сообщение от 40 до 60 байтов может занимать от 1 до 1,5 с для передачи. Это может быть проблематично при передвижении транспортных средств по шоссе. Базовая станция также требует **неизменности линейного времени (LTI)**, что означает, что радиоволна не должна изменять время прохождения пути;
- точность геолокации составляет около 100 м. Используя измерения силы сигнала RSSI или измерения времени пролета, можно получить опреде-

ленную степень точности. Лучшим решением является то, что три базовые станции триангулируют узел. Больше базовых станций может повысить точность.

Sigfox

Sigfox – это узкополосный протокол LPWAN (как NB-IOT), разработанный в 2009 г. в Тулузе, Франция. Учредительная компания имеет одноименное название. Это еще одна технология LPWAN, использующая нелицензированные полосы ISM для проприетарного протокола. У Sigfox есть некоторые черты, которые значительно сужают его использование:

- до 140 сообщений на устройство ежедневно по восходящей линии связи (рабочий цикл 1%, 6 сообщений в час);
- размер полезной нагрузки составляет 12 байтов для каждого сообщения (восходящая линия связи) и 8 байтов (нисходящая линия связи);
- пропускная способность до 100 бит/с для восходящей линии связи и 600 бит/с по нисходящей линии связи.

Первоначально Sigfox был однонаправлен и предназначен как чистая сенсорная сеть. Это означает, что поддерживается только связь с восходящей линией датчика. С тех пор стал доступен канал нисходящей линии связи.

Sigfox – запатентованная и закрытая технология. Хотя их оборудование открыто, однако в сеть не нужно и не должно подписываться. Партнеры по оборудованию Sigfox включают Atmel, TI, Silicon Labs и другие. Sigfox строит и управляет своей сетевой инфраструктурой, подобной расположению несущих LTE. Это совсем другая модель, отличная от LoRaWAN. LoRaWAN требует использования собственной сети PHY в своей сети, в то время как Sigfox использует несколько аппаратных производителей, но одну управляемую сетевую инфраструктуру. Sigfox вычисляет ставки по количеству устройств, подключенных к сетевой подписке клиента, профилю трафика на устройство и продолжительности контракта.

Хотя существуют строгие ограничения Sigfox с точки зрения пропускной способности и использования, он предназначен для систем, отправляющих небольшие и нечастые всплески данных. Кандидатами могут быть устройства IoT, такие как системы сигнализации, простые измерители мощности и датчики окружающей среды. Данные для различных датчиков обычно могут вписываться в ограничения (например, данные о температуре/влажности 2 байта при точности 0,004°). Нужно быть осторожным с той степенью точности, которую предоставляет датчик, и количеством данных, которые могут быть переданы. Один трюк для использования – данные состояния; состояние или событие могут просто быть сообщением без какой-либо полезной нагрузки. В этом случае протокол потребляет 0 байт. Хотя это не устраняет ограничений в вещании, это можно использовать для оптимизации питания.

Физический уровень Sigfox

Как уже упоминалось ранее, Sigfox – **ультраузкая полоса (UNB)**. Как следует из названия, передача использует очень узкий канал для связи. Вместо того, что-

бы распространять энергию по широкому каналу, узкое использование этой энергии ограничивается такими полосами:

- 868 МГц – Европа (правила ETSI 300-200);
- 902 МГц – Северная Америка (правила 15 части FCC).

В некоторых регионах, таких как Япония, существуют жесткие ограничения спектральной плотности, которые в настоящее время затрудняют развертывание сверхтонкой полосы.

Полоса шириной 100 Гц и использует **спектр ортогонального распространения последовательности (OSSS)** для сигнала восходящей линии связи и 600 Гц с использованием **гауссовой сдвиговой синхронизации частоты (GFSK)** для нисходящей линии связи. Sigfox отправит короткий пакет на случайный канал со случайной временной задержкой (от 500 до 525 мс). Этот тип кодирования называется **случайной частотой и множественным доступом с временным разделением (RFTDMA)**. Sigfox, как заявлено, имеет строгие параметры использования, в частности ограничения размера данных. В табл. 7.7 показаны эти параметры для каналов восходящей и нисходящей линии связи.

Таблица 7.7. Параметры каналов восходящей и нисходящей линии связи

	Восходящая связь	**Нисходящая связь**
Предел полезной нагрузки (байты)	12	8
Пропускная способность (бит/с)	100	600
Максимальное количество сообщений в день	140	4
Схема модуляции	DBPSK	GFSK
Чувствительность (дБм)	< 14	< 27

Двунаправленная связь является важной характеристикой протокола Sigfox и других протоколов. Однако двунаправленная связь в Sigfox требует некоторого объяснения. Режим пассивного приема отсутствует, что означает, что базовая станция не может просто отправить сообщение на устройство оконечной точки в любое время. Окно приема открывается для связи только после завершения окна передачи. Окно приема открывается только после 20-секундного периода, когда первое сообщение было отправлено узлом оконечной точки. Окно останется открытым в течение 25 с, что позволит получить короткое (4 байта) сообщение от базовой станции.

В Sigfox используются 333 канала шириной 100 Гц. Чувствительность приемника составляет –120 дБм/–142 дБм. Частотный скачок поддерживается с использованием псевдослучайного метода 3-х из 333 каналов. Наконец, мощность передачи указана как +14 дБм и +22 дБм в Северной Америке (рис. 7.12).

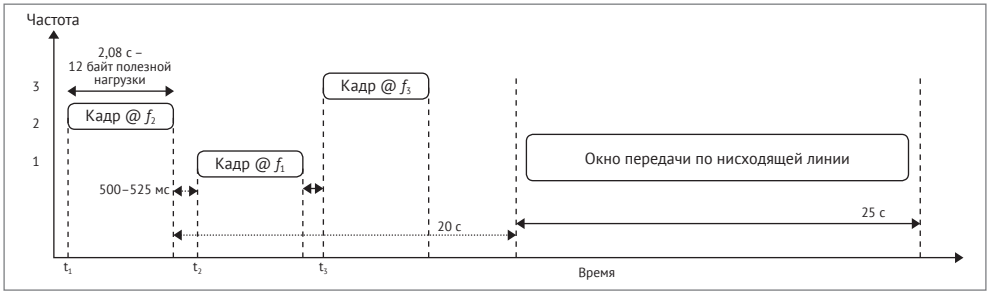


Рис. 7.12 ❖ Временная шкала передачи Sigfox.
Три копии полезной нагрузки передаются на трех уникальных случайных частотах с различными временными задержками. Окно передачи по нисходящей линии открывается только после последней восходящей линии связи

Уровень MAC Sigfox

Каждое устройство в сети Sigfox имеет уникальный идентификатор Sigfox. Идентификатор используется для маршрутизации и подписывания сообщений. Идентификатор используется для аутентификации устройства Sigfox. Еще одна характеристика связи Sigfox заключается в том, что он действует как *«сжечь и забыть»*. Получатели не подтверждают сообщения. Скорее, сообщение отправляется три раза на трех разных частотах в три разных направления узлом. Это помогает обеспечить целостность передачи сообщения. Модель *«сжечь и забыть»* не имеет возможности гарантировать, что сообщение действительно получено, поэтому передатчик должен делать как можно больше, чтобы обеспечить точную передачу (рис. 7.13).

Восходящий кадр Sigfox MAC

32 бита	16 бит	32 бита	От 0 до 96 бит	Варьируется	16 бит
Преамбула	Синхронизация кадра	ИД адресата	Полезная нагрузка	Аутентификация	Контрольная сумма

Нисходящий кадр Sigfox MAC

32 бита	13 бит	2 бита	8 бит	16 бит	Варьируется	От 0 до 64 бит
Преамбула	Синхронизация кадра	Флаги	Контрольная сумма	Аутентификация	Коды ошибок	Полезная нагрузка

Рис. 7.13 ❖ Структура пакета кадров Sigfox MAC
для восходящей и нисходящей линии связи

Кадры содержат преамбулу предопределенных символов, используемых для синхронизации при передаче. Поля синхронизации кадра определяют типы передаваемых кадров. FCS – это **контрольная последовательность кадров (FCS)**, используемая для обнаружения ошибок.

Пакет не содержит адрес назначения или другой узел. Все данные будут отправляться различными шлюзами в облачную службу Sigfox.

Предел данных можно понять и смоделировать из формата пакета уровня MAC:

$$\frac{\text{~200 бит восходящий пакет}}{\text{100 бит/с}} = 2 \text{ с.}$$

Поскольку каждый пакет передается три раза и мы знаем, что европейские правила (ETSI) ограничивают передачу в 1% рабочего цикла, мы можем рассчитать количество сообщений в час, используя максимальный размер полезной нагрузки в 12 байт:

$$\text{3600 с @ 1\% рабочего цикла} = \frac{\text{36 с передается сообщение}}{\text{ч}} \times \frac{\text{сообщение}}{\text{3 повтора} \times \text{2 с}} = \frac{\text{6 сообщений}}{\text{ч}}.$$

Хотя 12 байт является пределом полезной нагрузки, это сообщение может занять одну секунду для передачи. Ранние версии Sigfox были однонаправленными, но протокол теперь поддерживает двунаправленную связь.

Стек протокола Sigfox

Стек протокола аналогичен другим стекам, следующим модели OSI (табл. 7.8). Существует три уровня, подробно описанные здесь:

- **уровень PHY** – синтезирует и модулирует сигналы с использованием DBPSK в направлении восходящей линии связи и GFSK в направлении нисходящей линии связи, как описано ранее;
- **уровень MAC** – добавляет поля для идентификации/аутентификации устройства (HMAC) и кода исправления ошибок (CRC). Sigfox MAC не предоставляет никакой сигнализации. Это означает, что устройства не синхронизируются с сетью;
- **уровень кадра** – генерирует радиокадр из данных приложения. Кроме того, систематически прикрепляет порядковый номер к кадру.

Таблица 7.8. Стек протокола Sigfox и упрощенная модель OSI

Стек протокола Sygfox	Упрощенная модель OSI
Прикладной уровень	7. Прикладной уровень
	6. Представительский уровень
	5. Прикладной уровень
Frame	4. Транспортный уровень
	3. Сетевой уровень
MAC Layer	2. Канальный уровень
PHY Layer (868 МГц / 902 МГц Radios)	1. Физический уровень

Что касается безопасности, сообщения не шифруются ни в каком месте стека протоколов Sigfox. Клиент должен предоставить схему шифрования для данных полезной нагрузки (уровень 7 или 6 модели OSI). Никакие ключи не обмениваются через сеть Sigfox, однако каждое сообщение подписывается ключом, уникальным для идентификации устройства.

Топология Sigfox

Сеть Sigfox может быть плотной, как один миллион узлов на базовую станцию (рис. 7.14). Плотность – это функция количества сообщений, отправленных сетевыми устройствами. Все узлы, которые присоединяются к базовой станции, образуют звездную сеть.

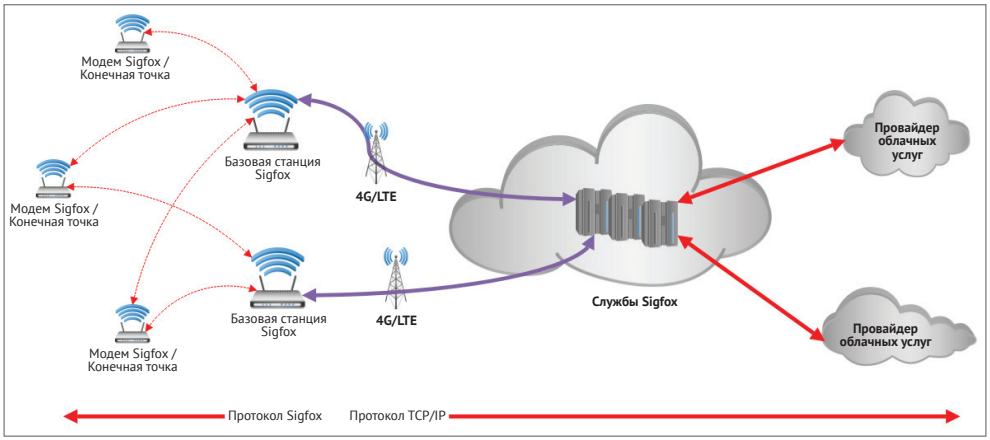


Рис. 7.14 ❖ Топология Sigfox. Sigfox использует свой собственный протокол, отличный от IP, и объединяет данные в сетевой сервер Sigfox, как данные IP

Все данные управляются через бэкэнд-сеть Sigfox. Все сообщения с базовой станции Sigfox должны поступать на внутренний сервер через IP-соединение. Служба облачных вычислений Sigfox является единственным местом назначения пакета. Бэкэнд будет хранить и отправлять сообщение клиенту после его аутентификации и проверки отсутствия дубликатов. Если данные должны быть переданы узлу оконечной точки, бэкенд-сервер будет выбирать шлюз с наилучшим подключением к оконечной точке и пересылать сообщение по нисходящей линии. Бэкэнд уже идентифицировал устройство по идентификатору пакета, и предварительное предоставление данных заставит данные отправлять в оконечный пункт назначения. В архитектуре Sigfox невозможно напрямую получить доступ к устройству. Ни бэкэнд, ни базовая станция не будут напрямую подключаться к оконечному устройству.

Бэкэнд – это также администрирование, лицензирование и предоставление услуг для клиентов. Облако Sigfox передает данные в пункт назначения, выбранный клиентом. Облачный сервис предлагает API с возможностью вставки для интеграции облачных функций Sigfox на сторонней платформе. Устройства могут быть зарегистрированы через другую облачную службу. Sigfox также предлагает обратные вызовы для других облачных сервисов. Это предпочтительный метод для извлечения данных.

Чтобы помочь обеспечить целостность данных в забываемой модели связи, несколько шлюзов могут получать передачу от узла; все последующие сообщения будут перенаправлены на бэкэнд Sigfox, и дубликаты будут удалены. Это добавляет уровень избыточности к приему данных.

Прикрепление узла оконечной точки Sigfox предназначено для облегчения установки. Отсутствует спаривание или сигнализация.

Заключение

Несмотря на некоторую общность между типами технологий связи на большие расстояния, они также нацелены на различные варианты использования и сегменты. Архитектор IoT должен мудро выбирать, какую дальнюю систему они намереваются принять. Как и другие компоненты системы IoT, LPWAN трудно изменить после развертывания. Соображения по выбору правильного LPWAN включают вопросы:

- Какую скорость передачи данных необходимо использовать для развертывания IoT?
- Может ли решение масштабироваться с тем же LPWAN по регионам? Есть ли надлежащий охват или нужно его построить?
- Какой диапазон передачи подходит?
- Существует ли какое-либо требование латентности для этого решения IoT? Может ли решение работать с очень высокими (несколькими секундами) задержками?
- Являются ли оконечные точки IoT питаемыми от батареи и какова стоимость их обслуживания?
- Каковы ограничения затрат на оконечные точки?

Существует еще несколько технологий LPWAN, таких как Ingenu в США и невесомые технологии (например, Weight-N, Weightless-W и Weightless-P). Невесомый подход интересен, так как это единственный LPWAN, работающий в телевизионном белом пространстве. MuLTEFire – это новая технология, использующая нелицензированные полосы частот, разработанные и продвигаемые Qualcomm и Nokia. Он предоставляет сотовые услуги LTE с использованием Wi-Fi, что позволяет частному оператору LTE развертывать частную сотовую сеть.

Для справки в табл. 7.9 описаны сходства и различия между протоколами LPWAN, описанными в этой главе.

Таблица 7.9. Сходства и различия между протоколами LPWAN

Спецификация	Cat-0 (LTEM) релиз 12	Cat-1 релиз 8	Cat-M1 релиз 13	Cat-NB релиз 13	LoRa/LoRaWAN	Sigfox
Полосы ISM	Нет	Нет	Нет	Нет	Да	Да
Общая полоса пропускания	20 МГц	20 МГц	1,4 МГц	180 КГц	125 КГц	100 КГц
Пиковая нагрузка на нисходящем направлении	1 Мб/с	10 Мб/с	1 Мб/с или 375 Кб/с	200 Кб/с	0,3–5 Кб/с	100 б/с
Пиковая нагрузка на восходящем направлении	1 Мб/с	5 Мб/с	1 Мб/с или 375 Кб/с	200 Кб/с	5 Кб/с до 5 Кб/с адаптивно	600 б/с
Дальность	LTE	LTE	~4x Cat-1	~7xCat-1	5 км в городе, 15 км в пригороде	До 50 км
Maximum Coupling Loss (MCL)	142,7 дБ	142,7 дБ	155,7 дБ	164 дБ	165 дБ	168 дБ
Энергопотребление в спящем режиме	Низкое	Высокое: ~2 мА	Очень низкое: ~15 мкА	Очень низкое: ~15 мкА	Экстремально низкое: 1,5 мкА	Экстремально низкое: 1,5 мкА
Конфигурация дуплекса	Полу/полный	Полный	Полу/полный	Полный	Полный	Полный
Антенны (MIMO)	1	2 MIMO	1	1	1	1
Задержка	50–100 мс	50–100 мс	10–15 мс	1,6–10 мс	500 мс – 2 с	До 60 с
Мощность передачи (UE)	23 дБ	23 дБ	20 дБ	23 дБ	14 дБ	14 дБ
Сложность разработки	50% от Cat-1	Сложно	25% от Cat-1	10% от Cat-1	Несложно	Несложно
Стоимость (относительная)	~ $15	~$30	~$10	~$5	~$15	~$3
Мобильность	Мобильный	Мобильный	Мобильный	Ограниченно	Мобильный	Ограниченно

Пройдя передачу данных от датчика и дальше, через архитектуру PAN и WAN, настало время обсудить направляемость и обработку данных IoT. Следующая глава – это первая обработка этих данных для их упаковки, обеспечения безопасности и маршрутизирования в нужное место. Это местоположение может быть узлом на краю или в тумане или облаком. Мы обсудим роль шлюза в обеспечении сетевого моста и возможности обработки данных на краю. Мы также подробно рассмотрим тип протоколов связи на основе IP, таких как MQTT и CoAP, необходимые для потоковой передачи данных от края до облака. В последующих главах рассказывается о поглощении, хранении и анализе данных, генерируемых IoT.

Глава 8

Маршрутизаторы и шлюзы

Интернет вещей имеет, скорее, отраслевую и экономическую направленность из-за количества устройств, которые в конечном итоге будут развернуты, и объема данных, которые эти устройства будут производить. Существует два метода относительно того, как будет формироваться IoT:

- граничные, или краевые, датчики и устройства обеспечат прямой путь к облаку. Это означает, что эти узлы и датчики граничного уровня будут иметь достаточно ресурсов, аппаратных средств, программного обеспечения и соглашений об уровне обслуживания для прямой передачи данных через WAN;
- датчики граничного уровня образуют группы и кластеры вокруг шлюзов и маршрутизаторов для обеспечения промежуточных областей, преобразования протоколов и возможностей обработки на границах / в туманах и будут управлять безопасностью и аутентификацией между датчиками и глобальной сетью.

Первая модель является сложной и дорогостоящей для мощных и недорогих датчиков/преобразователей/устройств граничного уровня. Более логичным будет второй вариант. Роль граничного маршрутизатора или шлюза для датчиков/устройств включает в себя формальные сетевые возможности, которые предоставляют современные маршрутизаторы, такие как маршрутизация каналов, переадресация портов, туннелирование, безопасность и резервирование. Принципы маршрутизации и ретрансляции TCP/IP описаны во многих источниках и слишком широки для изучения здесь. В этой главе рассматривается роль и необходимость маршрутизаторов граничного уровня, а также приведены советы и рекомендации относительно функций, которые следует учитывать при развертывании массового решения IoT.

Функции маршрутизации

В архитектуре IoT маршрутизатор играет значительную роль в общем управлении, масштабировании и безопасности системы. Часто бывает, что роль маршрутизатора упрощается, чтобы он действовал просто как шлюз из одного протокола в другой (преобразователь сотового к Bluetooth). Для коммерческого

или промышленного развертывания необходимо учитывать гораздо больше, особенно когда устройства находятся в удаленных и подвижных системах.

Функции шлюза

В этой книге мы рассмотрели несколько типов беспроводной связи, каждая из которых требует какой-либо формы передатчика и приемника, а также механизм для преобразования трафика для подключения к интернету. Это самая важная роль пограничного шлюза IoT. Независимо от того, является ли среда Bluetooth-сетью, требующей узла моста или eNodeB сотовой сети, роли аналогичны. Шлюз преобразует и направляет данные между двумя несходными сетями.

Маршрутизатор может быть шлюзом. Маршрутизаторы направляют и управляют трафиком между подобными сетями. В архитектуре IoT может быть установлена смесь 6LoWPAN, которая адресуется по IPv6, обмениваясь информацией с внешним миром через устройство, которое действует как шлюз, соединяющий протоколы 802.15.4 с физическим транспортом Wi-Fi или 802.3, но также направляя и маршрутизируя данные с использованием IP-уровня, разделяемого между интернетом и mesh-сетью 6LoWPAN.

Часто пограничный шлюз также является центральным контроллером PAN. Это означает, что все функции управления сетью PAN, обеспечение безопасности, аутентификация и предоставление новых узлов, данных управления и устройств управления питанием относятся к ответственности пограничного шлюза.

Маршрутизация

Основная функция маршрутизатора – это обеспечение соединений между сегментами сети. Маршрутизация считается функцией третьего уровня стандартной модели OSI, поскольку она использует уровень IP-адресации для управления передачей пакетов. По сути, все маршрутизаторы полагаются на таблицу маршрутизации для управления потоком данных. Таблица маршрутизации используется для поиска наилучшего соответствия IP-адресу назначения пакета.

Существует несколько проверенных алгоритмов, используемых для эффективной маршрутизации. Одним типом маршрутизации является динамическая маршрутизация, где алгоритмы реагируют на изменения в сети и топологии. Информация о состоянии сети распространяется протоколом маршрутизации по времени или по случившемуся обновлению. Примерами динамической маршрутизации являются маршрутизация вектора расстояния и маршрутизация состояния канала. В качестве альтернативы статическая маршрутизация важна и полезна для небольших сетей, которым нужны определенные пути, настроенные между маршрутизаторами. Статические маршруты неадаптивны, поэтому нет необходимости сканировать топологию или обновлять метрики. Они предустановлены на маршрутизаторе:

- **маршрутизация по самому короткому пути** – построение графа, представляющего маршрутизаторы в сети. Дуга между узлами представ-

ляет собой известную связь или соединение. Алгоритм просто находит кратчайший путь из любого источника к любому пункту назначения;

- **лавинная маршрутизация** – каждый пакет повторяется и транслируется каждым маршрутизатором в каждую конечную точку его связей. Это генерирует огромное количество дублирующих пакетов и требует, чтобы в заголовке пакета был установлен счетчик переходов, чтобы гарантировать, что пакеты имеют ограниченное время жизни. Альтернативой является выборочная рассылка, которая наводняет сеть только в основном направлении пункта назначения. Лавинные сети являются основой сетей Bluetooth;
- **маршрутизация на основе потока** – проверяет текущий поток в сети до определения пути. Для любого данного соединения, если известны пропускная способность и средний поток, вычисляется средняя задержка пакета по этому соединению. Этот алгоритм находит минимальное среднее значение;
- **маршрутизация на основе вектора расстояния** – таблица маршрутизаторов содержит *самое известное расстояние* до каждого пункта назначения. Таблицы обновляются соседними маршрутизаторами. Таблица содержит запись для каждого маршрутизатора в подсети. Каждая запись содержит предпочтительный маршрут/путь и приблизительное ожидаемое расстояние до пункта назначения. Расстояние может быть метрикой количества переходов, задержки или длины очереди;
- **маршрутизация по состоянию канала** – маршрутизатор сначала обнаруживает всех своих соседей с помощью специального пакета HELLO. Маршрутизатор измеряет задержку для каждого из своих соседей путем отправки пакета ECHO. Эта информация о топологии и времени затем рассылается всем маршрутизаторам в подсети. Полная топология строится и публикуется всеми маршрутизаторами;
- **иерархическая маршрутизация** – маршрутизаторы разделены на регионы и имеют иерархическую топологию. Каждый маршрутизатор поддерживает понимание своего региона, но не всей подсети. Иерархическая маршрутизация также является эффективным средством управления размером таблицы маршрутизации и ресурсами в ограниченных устройствах;
- **широковещательная маршрутизация** – каждый пакет содержит список адресов назначения. Широковещательный маршрутизатор исследует адреса и определяет набор выходных линий для передачи пакета. Маршрутизатор будет генерировать новый пакет для каждой выходной линии и включать только адресаты, необходимые для этого вновь сформированного пакета;
- **маршрутизация с помощью многоадресной рассылки**: сеть разделена на четко определенные группы. Приложение может отправлять пакет всей группе, а не одному адресату или на широковещательный адрес.

Важной метрикой в маршрутизации является время сходимости. Конвергенция происходит, когда все маршрутизаторы в сети имеют одну и ту же топологическую информацию и состояние.

Типичные граничные маршрутизаторы будут поддерживать такие протоколы маршрутизации, как **Border Gateway Protocol** (**BGP**), **Open Shortest Path First** (**OSPF**), **Routing Information Protocol** (**RIP**) и RIPng. Архитектор, использующий граничные маршрутизаторы в реальной работе, должен знать о перегрузке и стоимости использования определенного протокола маршрутизации по сравнению с другими, особенно если соединение между маршрутизаторами является WAN-соединением с ограничением данных:

- **BGP** – BGP-4 является стандартом для протоколов интернет-доменной маршрутизации и описан в RFC 1771; он используется большинством интернет-провайдеров. BGP – это алгоритм динамической маршрутизации на основе вектора расстояния, он рекламирует целые пути в сообщениях обновлений маршрутизации. Если таблицы маршрутизации являются большими, это требует значительной пропускной способности. BGP отправляет 19-байтовое сообщение keepalive каждые 60 с для поддержания соединения. BGP может быть слабым протоколом маршрутизации для топологии mesh-сети, поскольку BGP поддерживает соединение с соседями. BGP также страдает от роста таблиц маршрутизации в больших топологиях. BGP также уникален, так как это один из единственных протоколов маршрутизации на основе TCP-пакетов;
- **OSPF** – этот протокол описан в RFC 2328, он обеспечивает преимущества масштабирования сети и сходимости. Интернет-магистраль и корпоративные сети интенсивно используют OSPF. OSPF – это алгоритм состояния соединения, который поддерживает IPv4 и IPV6 (RFC 5340) и работает с IP-пакетами. Преимущество заключается в обнаружении за секунды динамических изменений соединений и реагировании;
- **RIP** – вторая версия RIP представляет собой алгоритм маршрутизации вектора расстояния, основанный на подсчете переходов с использованием протокола внутреннего шлюза. Первоначально основанный на алгоритме Беллмана-Форда, теперь он поддерживает подсети с изменяющимися размерами, преодолевая ограничения исходной версии. Петли в таблице маршрутизации ограничены заданием максимального количества переходов в пути (15). RIP работает по UDP и поддерживает только трафик IPv4. RIP имеет более длительное время сходимости, чем такие протоколы, как OSPF, но его легко администрировать для топологий небольшого граничного маршрутизатора. Тем не менее, сходимость для RIP со всего несколькими маршрутизаторами может занять несколько минут;
- **RIPng** – RIPng означает RIP следующего поколения (RFC 2080). Он позволяет поддерживать трафик IPv6 и IPsec для аутентификации.

Типичная таблица маршрутизации, используемая в производственной среде таким маршрутизатором, как Cradlepoint IBR900, выглядит следующим об-

разом:

```
[administrator@IBR900-e11: /]$ route
Table: wan
Destination        Gateway    Device UID       Flags    Metric   Type
default            96.19.152.1  wan            onlink   0        unicast

Table: main
Destination        Gateway    Device UID       Flags    Metric   Type
96.19.152.0/21     *            wan                    0        unicast
172.86.160.0/20    *            iface:pertino0         0        unicast
172.86.160.0/20    None         None                   256      blackhole
192.168.1.0/24     *            primarylan             0        unicast
2001:470:813b::/48 *            *iface:pertino0        256      unicast
fe80::/64          *            lan                    256      unicast

Table: local
Destination       Gateway     Device UID       Flags    Metric   Type
96.19.152.0        *            wan                    0        broadcast
96.19.153.13       *            wan                    0        local
96.19.159.255      *            wan                    0        broadcast
127.0.0.0          *            *iface:lo              0        broadcast
.
.
.
```

В этом примере есть три таблицы: `wan`, `main` и `local`. Каждая таблица содержит конкретные пути маршрутизации, характерные для этого интерфейса:

- **Destination** – полный или частичный IP-адрес назначения пакета. Если таблица содержит IP, она будет ссылаться на остальную часть записи, чтобы разрешить интерфейс для маршрутизации на частичный адрес; может быть задана с префиксом /. Это указывает фиксированные позиции битов адреса для разрешения. Например,`/24` в `192.168.1.0/24` указывает, что старшие 24 бита 192.168.1 являются фиксированными, а младшие 8 бит могут иметь любой адрес в подсети `192.168.1. *`;
- **Gateway** – это интерфейс для направления пакетов, соответствующий найденному адресату. В предыдущем случае шлюз указан как `96.19.152.1`, а пункт назначения – `default`. Это означает, что исходящий WAN-адрес `96.19.152.1` будет использоваться для всех адресов назначения. Это, по сути, пересылка IP;
- **DeviceUID** – это буквенно-цифровой идентификатор интерфейса для отправки данных. Например, любое место назначения в подсети `172.19.152.0/21` направит пакеты в интерфейс под названием `iface: pertino0`. Часто это поле будет выражаться цифровым IP-адресом, а не символической ссылкой;
- **Flags** – используется для диагностики и указания состояния маршрута. Состояния могут быть «маршрут работает, использовать шлюз»;
- **Metric** – это расстояние до места назначения, обычно подсчитывается по количеству переходов;

- **Type** – могут использоваться несколько типов маршрутов:
  - `unicast`: маршрут является реальным путем к месту назначения;
  - `unreachable`: цель недоступна. Пакеты отбрасываются, и генерируется сообщение ICMP, указывающее, что хост недоступен. Местный отправитель получает ошибку `EHOSTUNREACH`;
  - `blackhole`: получатели недостижимы. В отличие от типов «prohibit», пакеты будут тихо отброшены. Локальные отправители получают ошибку `EINVAL`;
  - `prohibit`: получатели недостижимы. Пакеты будут отбрасываться и генерируются ICMP-сообщения. Локальные отправители получают ошибку `EACCES`;
  - `local`: получателем является этот хост. Пакеты заворачиваются обратно и доставляются локально;
  - `broadcast`: пакеты будут передаваться как широковещательные через интерфейс ко всем получателям4
  - `throw`: специальный маршрут управления, чтобы насильно отбросить пакеты и сгенерировать сообщения ICMP о недоступности.

В предыдущем примере также следует отметить, что адреса IPv6 смешиваются с адресами IPv4. Например, `2001:470:813b::/48` в основной таблице – адрес IPv6 с `/48`-битной подсетью.

Отказоустойчивость и внеполосное управление

Отказоустойчивость критична для некоторых граничных маршрутизаторов IoT, особенно для приложений для транспортных средств и по уходу за пациентами. Отказоустойчивость, как следует из названия, – это переключение с одного интерфейса WAN на другой, когда основной источник потерян. Потеря WAN может быть связана с потерей сотовой связи в туннеле. Компании-перевозчику с автопарком передвижных холодильных камер может потребоваться гарантированное подключение, при том что услуги сотовой связи меняются по всей стране. Переход от одного сотового оператора к другому с использованием нескольких идентификаторов SIM-карты может помочь смягчить и сгладить переподключение. Другим вариантом может быть использование клиентского Wi-Fi в качестве основного интерфейса WAN для внутреннего мониторинга состояния, но подключение к сотовой WAN ради отказоустойчивости, если сигнал Wi-Fi потерян. Отказоустойчивость должна быть безболезненной и автоматической без потери пакетов или заметного влияния на задержку данных.

Внеполосное управление (OOBM) также должно рассматриваться для устройств IoT. OOBM полезен в отказоустойчивых условиях, когда для управления оборудованием необходим выделенный и изолированный канал. Иногда это называется **управлением лампочками (LOM)**, если первичные системы вышли из строя, повреждены или имеют потерю мощности; по-прежнему возможно управлять и проверять оборудование удаленно через канал боковой полосы. В IoT это может быть полезно для ситуаций, требующих гарантирован-

ного времени безотказной работы и дистанционного управления, таких как мониторинг нефти и газа или промышленная автоматизация. Хорошо продуманная система OOBM не должна иметь никакого отношения к контролируемой системе для своего функционирования. Типичные способы управления, такие как туннели VNC или SSH, требуют, чтобы устройство было загружено и функционировало.

OOBM должен быть добавочным и изолированным от системы, как показано на рис. 8.1.

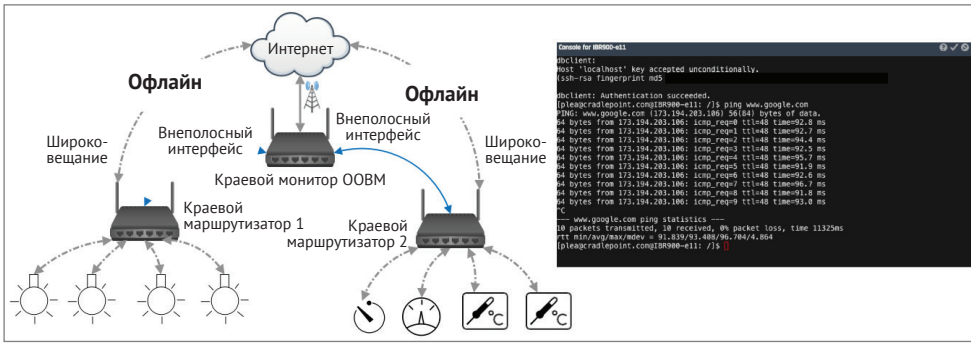


Рис. 8.1 ❖ Пример конфигурации для внеполосного управления

VLAN

VLAN функционирует, как любая другая физическая локальная сеть, но позволяет группировать компьютеры и другие устройства, даже если они физически не привязаны к одному сетевому коммутатору. Разделение происходит на уровне линии передачи данных (второй уровень) модели OSI. VLAN – это форма сетевой сегментации устройств, приложений или пользователей, хотя они находятся в одной и той же физической сети. VLAN также может группировать хосты вместе, хотя они не находятся на одном сетевом коммутаторе, что существенно облегчает разбиение на сети без использования дополнительных кабелей. Стандарт IEEE 802.1Q – это стандарт, по которому построены VLAN. По сути, VLAN использует идентификатор или тег, состоящий из 12 бит в кадре Ethernet. Таким образом, существует жесткое ограничение в 4096 потенциальных VLAN в одной физической сети.

Коммутатор может назначить порт для непосредственной привязки к определенной VLAN. Поскольку VLAN работает на втором уровне стека, трафик может быть туннелирован через третий уровень, что позволяет географически разделенным VLAN использовать общую топологию (рис. 8.2).

Выше показана корпоративная точка продажи (POS) и система VOIP, которая фактически изолирована от набора устройств IoT, а также гостевого Wi-Fi. Это делается посредством адресации VLAN, хотя система использует одну и ту же физическую сеть. Здесь мы предполагаем, что это интеллектуальное решение

IoT, где все граничные устройства и датчики IoT поддерживают IP-стек и адресуются через локальную сеть.

Дизайн VLAN особенно полезен в мире IoT. Изолирование устройств IoT от других корпоративных функций является типичным сценарием. VLAN полезны только при работе с устройствами, поддерживающими IP-адресацию.

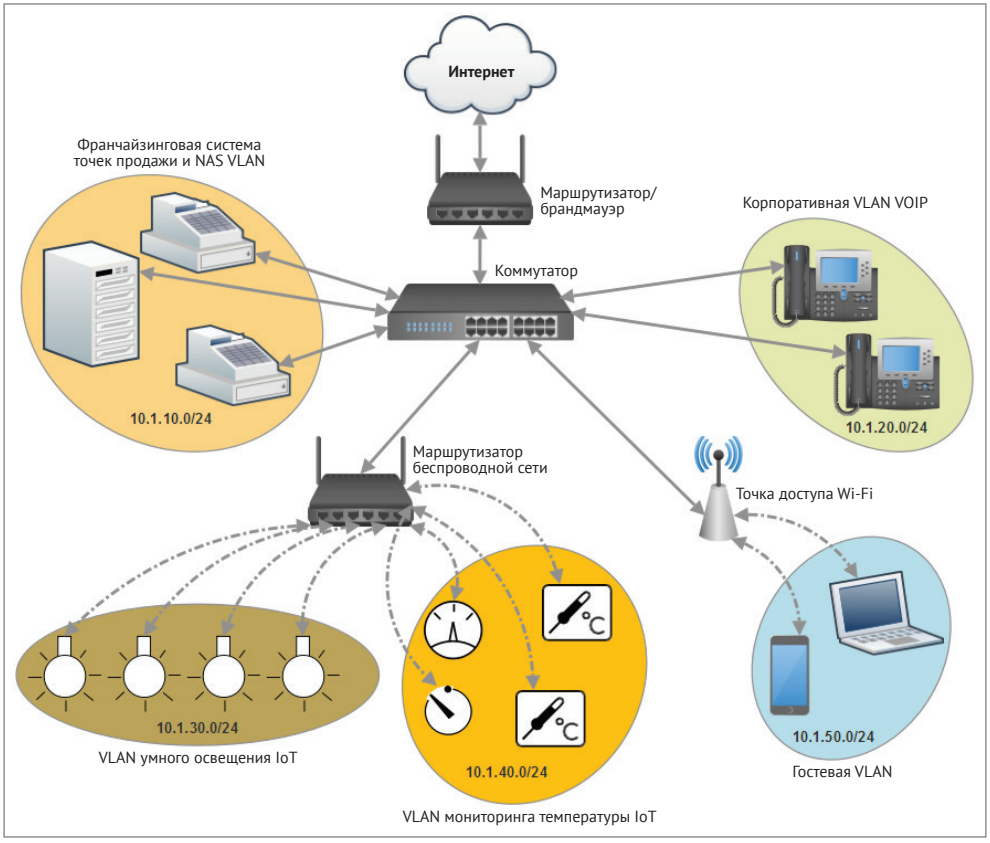


Рис. 8.2 ❖ Пример архитектуры VLAN в сценарии филиалов или розничной торговли

VPN

VPN-туннели используются для установления безопасного подключения к удаленной сети через публичную сеть. Например, VPN-туннели могут использоваться через интернет для отдельного пользователя для подключения к защищенной корпоративной сети во время поездок или для соединения двух офисных сетей для работы в качестве одной сети. Две сети настраивают безопасное соединение через (обычно) незащищенный интернет, применяя протоколы шифрования VPN.

Для внедрения IoT для передачи данных с удаленных датчиков и периферийных устройств в корпоративную или частную локальную сеть необходима VPN. Как правило, корпорация будет находиться за брандмауэром, а VPN – это единственный способ перемещения данных непосредственно на частный локальный сервер. В этих случаях VPN может быть необходимым компонентом маршрутизатора для соединения сетей. Позже в этой главе будут рассмотрены программные защищенные сети как альтернативный метод обеспечения безопасности сетей.

Существует несколько VPN:

- **Internet Protocol Security (IPSec) VPN** – традиционная технология VPN, которая находится на сетевом уровне стека OSI и обеспечивает передачу данных через туннель между двумя конечными точками;
- **OpenVPN** – это VPN с открытым исходным кодом для безопасного соединения «точка-точка» и «место-место» в маршрутизированных или мостовых конфигурациях. Она включает специальный протокол безопасности, использующий SSL/TLS (OpenSSL) для обмена ключами и управления шифрованием и обмена данными. Она может работать через транспорты UDP и TCP. SSL распространен в большинстве приложений для браузеров, поэтому SSL VPN-системы могут обеспечивать защищенные туннели на для приложений, а не всей сети;
- **Generic Routing Encapsulation (GRE)** – создает соединения «точка-точка» между конечными точками через туннель, аналогичный туннелю VPN, но инкапсулирует его полезную нагрузку. Он обертывает внутренний пакет во внешний пакет. Это позволяет передавать данные полезной нагрузки через другие IP-маршрутизаторы и туннели неизмененными. Кроме того, туннели GRE могут транспортировать IPV6 и многоадресные передачи;
- **Layer 2 Tunneling Protocol (L2TP)** – создает соединение между двумя частными сетями через дейтаграммы UDP, типично используемыми для VPN или как часть служб доставки провайдерами. В протоколе нет встроенной безопасности или шифрования, и для этого часто используется IPsec.

VPN должен либо доверять базовым сетевым протоколам, либо обеспечивать собственную безопасность. VPN-туннели обычно используют IPsec для аутентификации и шифрования пакетов, передаваемых по туннелям. Чтобы настроить туннельный маршрутизатор VPN на одном конце, должно быть другое устройство (обычно это маршрутизатор), которое также поддерживает IPsec на другом конце. **Internet Key Exchange (IKE)** – это протокол безопасности в IPsec. IKE имеет две фазы. Первая фаза отвечает за установление безопасного канала связи, а на втором этапе установленный канал используется партнерами IKE. Маршрутизатор имеет несколько разных параметров протокола безопасности для каждой фазы, но выбор по умолчанию будет достаточным для большинства пользователей. Каждый обмен IKE использует один алгоритм шифрования, одну хеш-функцию и одну группу DH для безопасного обмена:

- **Encryption** – используется для шифрования сообщений, отправляемых и получаемых по IPsec. Типичные стандарты и алгоритмы шифрования включают AES 128, AES 256, DES и 3DES;
- **Hash** – используется для сравнения, аутентификации и проверки данных по VPN, обеспечения того, что они приходят в правильной форме, и для получения ключей, используемых IPSec. Типичные функции хеша, которые следует ожидать в маршрутизаторе корпоративного уровня, включают MD5, SHA1, SHA2 256, SHA2 384 и SHA2 512. Обратите внимание, что некоторые комбинации шифрования/хеша (такие как 3DES с SHA2 384/512) являются дорогостоящими при вычислении, что влияет на производительность WAN. AES обеспечивает хорошее шифрование и работает намного быстрее, чем 3DES;
- **группы DH**: группа DH (Diffie-Hellman) является свойством IKE и используется для определения длины простых чисел, связанных с генерацией ключей. Надежность генерируемого ключа частично определяется надежностью группы DH. Группа 5, например, обладает большей прочностью, чем группа 2:
  - **группа 1**: 768-битный ключ;
  - **группа 2**: 1024-битный ключ;
  - **группа 5**: 1536-битный ключ.

В первой фазе IKE вы можете выбрать одну группу DH, только если используете агрессивный режим обмена.

Алгоритмы перечислены в порядке приоритета. Вы можете изменить порядок этого списка приоритетов, щелкнув и перетащив алгоритмы вверх или вниз. Любой выбранный алгоритм может использоваться для IKE, но алгоритмы в верхней части списка, вероятно, будут использоваться чаще.

Предупреждение по поводу мобильных и зависимых от питания развертываниях IoT. Традиционная VPN не может противостоять перемещению и выходу из постоянного сетевого соединения (например, сотового роуминга, переключения несущей или устройств со случайным питанием). Если сетевой туннель нарушен, это вызывает тайм-ауты, разъединения и сбои. Некоторое мобильное программное обеспечение VPN, такое как **Host Identity Protocol (HIP)** из IETF, пытается решить проблему, отключив различные IP-адреса, используемые при роуминге в VPN-логическое соединение. Другой альтернативой является **Software-Defined Networking (SDN)**, которое будет рассмотрено далее в этой книге.

Управление скоростью трафика и QoS

Функции управления скоростью трафика и качества услуг (QoS) полезны при развертывании, требующем гарантированного уровня обслуживания при работе с перегрузками или переменными нагрузками на сеть. Например, в случае использования IoT при смешивании живых видеопотоков и публичного Wi-Fi, видеопотоки могут нуждаться в приоритизации и гарантированном уровне качества, особенно для общественной безопасности или наблюдения.

Оставленные, как есть, поступающие данные от глобальной сети к пограничному маршрутизатору будут обслуживаться по принципу «первым пришел – первым обслужили»:

- **функции QoS** – позволяют администратору назначать уровни приоритета для заданного IP-адреса, размещенного на маршрутизаторе, или определенному порту. Функции QoS управляют только каналом восходящей линии связи. Они особенно полезны в тех случаях, когда канал восходящей линии связи имеет гораздо меньшую пропускную способность, чем нисходящая линия. Как правило, потребительский широкополосный доступ будет иметь что-то вроде восходящей линии 5 Мбит/с и нисходящей линии связи 100 Мбит/с, а QoS обеспечивает способ балансировки нагрузки на ограниченную восходящую линию. QoS не назначает жесткие лимиты и не сегментирует соединение, как это делает управление скоростью трафика;
- **функции управления скоростью трафика** – управление скоростью трафика – это статическая форма предопределения полосы пропускания. Например, связь 15 Мбит/с может быть разделена на меньшие сегменты по 5 Мбит/с. Эти сегменты должны быть назначены предварительно. Как правило, это лишние затраты, поскольку выделенные части полосы пропускания могут не освобождаться в случае необходимости;
- **динамическое управление скоростью и приоритет пакетов** – современные маршрутизаторы поддерживают атрибуты динамического управления скоростью. Это позволяет администратору динамически назначать правила сегментации полосы пропускания для входящего и исходящего трафика. Он также может управлять чувствительными к задержкам пакетами (например, видео или пользовательским интерфейсом) для приложений реального времени. Динамическое управление скоростью и приоритет пакетов позволяют создавать правила на основе типа данных или приложения, а не только IP-адреса или порта.

Метод классификации и управления сетевым трафиком – это **дифференцированные службы (DiffServ)**. DiffServ использует 6-битную кодовую точку дифференцированного сервиса (DSCP) в заголовке IP для классификации пакетов. Концепция DiffServ заключается в том, что сложные функции (такие как классификация пакетов и политики) могут выполняться на краю сети граничными маршрутизаторами, которые затем маркируют получаемый пакет, чтобы обеспечить определенное поведение при каждой передаче. Трафик, входящий в маршрутизатор DiffServ, подлежит классификации и приведению к требуемым условиям. Кроме того, маршрутизатор DiffServ может изменять классификацию пакета, ранее маркированного другим маршрутизатором. DiffServ – это «крупнозерновой» инструмент для управления трафиком, поскольку не все маршрутизаторы в цепочке нуждаются в его поддержке. Затем маршрутизатор будет управлять различными классами пакетов с помощью функций QoS. Альтернативно, IntServ, что означает Integrated Services, помогает в QoS и гарантирует, что все маршрутизаторы в цепочке поддерживают его. Это форма «мелкозернистого» QoS.

Другим аспектом качества сети является **среднее значение оценки (MOS)**. MOS – среднее арифметическое отдельных значений по шкале качества системы с точки зрения пользователя. Это обычно используется в приложениях **Голос по протоколу интернет (VOIP)**, но, безусловно, может использоваться для систем видеонаблюдения, обработки изображений, потоковой передачи данных и для удобства использования пользовательского интерфейса. Он основан на субъективном рейтинге от одного до пяти (где единица означает наихудшее качество, пять – наилучшее качество), и его следует использовать в петле обратной связи для увеличения емкости или уменьшения размеров данных для соответствия емкости.

Граничный маршрутизатор, соединяющий PAN с WAN на основе IP, имеет в своем распоряжении несколько вариантов, чтобы реагировать на изменения качества канала и ухудшения сетевых сервисов, например, в развертывании IoT для грузовых автопарков, где сигнал несущей может ухудшаться. В этих ситуациях маршрутизатор может использовать TCP **Performance Enhancing Proxies (PEP)** для преодоления и компенсации изменений качества (RFC 3135). PEP могут использоваться на транспортном уровне или прикладном уровне стека и различаются в зависимости от физического носителя. Формы PEP включают следующее:

- **ProxyPEP** – здесь прокси действует как посредник, который имитирует конечную точку;
- **DistributionPEP** – PEP может работать на одном конце соединения или на обоих (модель распределения).

PEP состоит из следующих функций:

- **разделить TCP** – PEP разбивает соединение на несколько сегментов, чтобы преодолеть большие задержки, влияющие на окна TCP. Это обычно используется в спутниковой связи;
- **фильтрация ACK** – в каналах, где скорости передачи данных не являются симметричными (например, Cat-1: 10 Мбит/с вниз, 5 Мбит/с вверх), фильтр ACK помогает, накапливая или сокращая TCP ACK для повышения производительности;
- **перехват** – это форма интегрированного прокси-сервера, используется для скрытия помех и коллизий по беспроводным каналам связи. Он перехватывает дубликаты ACK в сети, удаляет их и заменяет потерянным пакетом. Это предотвращает произвольное уменьшение размера окна TCP отправителем;
- **D-Proxy** – PEP для оказания помощи в беспроводных сетях путем распространения прокси-сервера TCP на каждую сторону соединения. Прокси контролируют порядковые номера пакетов данных TCP путем поиска потерянных пакетов. При обнаружении, прокси-серверы открывают временный буфер и улавливают пакеты, пока отсутствующий не будет восстановлен и повторно вставлен в последовательность.

Функции безопасности

Граничный маршрутизатор или шлюз выполняет еще одну важную задачу, обеспечивая уровень безопасности между WAN, интернетом и базовыми устройствами PAN/IoT. Многим устройствам не хватает необходимых ресурсов, памяти и вычислительной мощности для обеспечения надежности безопасности и обеспечения. Независимо от того, создает ли архитектор свой собственный

шлюзовой сервис или приобретает его, для защиты компонентов IoT следует учитывать следующий список свойств.

Защита с помощью брандмауэра – самая базовая форма безопасности. Существуют две основные формы брандмауэров для телекоммуникаций. Первый – это сетевой брандмауэр, который фильтрует и управляет потоком информации из одной сети в другую. Второй – это брандмауэр на базе хоста, который локально защищает приложения и службы на этой машине. В случае граничных маршрутизаторов IoT мы фокусируемся на сетевых брандмауэрах. По умолчанию брандмауэр предотвратит проникновение определенных типов сетевого трафика в зону, защищенную брандмауэром, но любой трафик, происходящий из этой зоны, может идти наружу. Брандмауэр найдет и изолирует информацию на основе пакетов, состояний или приложений в зависимости от сложности брандмауэра. Как правило, зоны создаются по сетевым интерфейсам с правилами, предназначенными для управления потоком трафика между зонами. Примером может служить граничный маршрутизатор с гостевой зоной Wi-Fi и корпоративной частной зоной.

Пакетный брандмауэр может изолировать и подавлять определенный трафик на основе IP-адреса источника или получателя, портов, MAC-адресов, IP-протоколов и другой информации, содержащейся в заголовке пакета. Брандмауэр с состоянием работает на четвертом уровне стека OSI. Он собирает и объединяет пакеты, ища шаблоны и информацию о состоянии, такую как новые соединения и существующие соединения. Фильтрация приложений еще сложнее, т. к. она может производить поиск по определенным сетевым потокам приложений, включая FTP-трафик или HTTP-данные.

Брандмауэр также может использовать **демилитаризованную зону (DMZ)**. DMZ – это просто логическая зона. Хост DMZ эффективно не защищен брандмауэром в том смысле, что любой компьютер в интернете может попытаться удаленно получить доступ к сетевым службам по IP-адресу DMZ. Типичные виды использования включают запуск общего веб-сервера и обмен файлами. Хост DMZ обычно задается прямым IP-адресом.

Переадресация портов – это концепция, позволяющая открывать определенные порты за брандмауэром. Для нескольких устройств IoT необходим открытый порт для предоставления сервисов, которые контролируются облачными компонентами. Опять же, построено правило, которое позволяет указанному IP-адресу в защищенной зоне брандмауэра иметь открытый порт.

Как DMZ, так и переадресация портов открывают порты и интерфейсы в вообще-то защищенной сети. Следует проявлять осторожность, чтобы это действительно было намерением архитектора. В массовом развертывании IoT с множеством граничных маршрутизаторов общее правило для открытия порта может быть полезно для одного местоположения, но для другого будет существенным риском для безопасности. Кроме того, проведение аудита DMZ и открытых портов должно быть процессом безопасности, поскольку сетевая топология и конфигурации меняются со временем. DMZ в любой момент может привести к открытой дыре в сетевой безопасности позже.

Метрики и аналитика

Во многих случаях граничные устройства IoT будут предметом соглашений уровня обслуживания (SLA) или ограничениями для данных при использовании измеряемых данных, как в случае 4G LTE. В других ситуациях граничный маршрутизатор или шлюз представляет собой множество других сетей PAN и устройств IoT. Он должен служить центральным (но местным) органом по здоровью сети/mesh-сети PAN. Метрики и аналитика полезны для сбора и решения проблем с подключением и стоимостью, особенно по мере увеличения масштабов IoT. Типичные метрики и сборки должны включать следующее:

- **аналитика времени бесперебойной работы WAN** – исторические тенденции, уровни обслуживания;
- **использование данных** – вход, выход, совокупность для каждого клиента, для каждого приложения;
- **полоса пропускания** – случайный или запланированный анализ полосы пропускания на входе и выходе;
- **здоровье PAN** – пропускная способность, аномальный трафик, реорганизация mesh-сети;
- **целостность сигнала** – сила сигнала, обзор места;
- **местоположение** – GPS-координаты, перемещение, изменение местоположения;
- **контроль доступа** – подключенные клиенты, вход администратора в систему, изменения конфигурации маршрутизатора, успех/неудача аутентификации PAN;
- **отказоустойчивость** – количество событий отказа, время и продолжительность.

Эти типы метрик должны собираться и контролироваться по расписанию и автоматически. Кроме того, продвинутые маршрутизаторы должны иметь возможность создавать правила и предупреждения, когда на границе видны определенные события или аномальное поведение.

Обработка на краю

Как мы увидим далее в главе, исследуя облачные и туманные вычисления, граничные маршрутизаторы способны предоставлять вычислительные ресурсы, близко к тому месту, где генерируются данные. Это важная характеристика в граничных маршрутизаторах, особенно для решений IoT. Поскольку больше решений IoT построено в удаленных и разнообразных местах, наличие возможности локальных вычислений важно во многих случаях использования. Вычисление на краю может воздействовать на пользовательские данные с помощью таких методов, как:

- фильтрация и агрегация;
- изменение данных;
- анализ безопасности и обнаружения вторжений;

- ключевой менеджмент;
- процессоры правил движков/событий;
- кэширование и хранение.

Граничные маршрутизаторы будут различаться по размеру и вычислительной мощности. Это не серверы центров обработки данных или стоечное оборудование. Как правило, характеристики граничного маршрутизатора будут такими, как показано в табл. 8.1.

Таблица 8.1. Примеры граничных маршрутизаторов

Свойство	Потребительский маршрутизатор	Граничный маршрутизатор среднего уровня	Туманный узел высокого уровня
Марка	Apple Airport Extreme	Advantech WISE-3310	HP Edgeline EL20 Gateway
MSRP	~ $199	~ $547	~ $1400
SOC/процессор Mfr.	Broadcom 53019	Freescale i.MX6	Intel 4300U
Тип и скорость ЦПУ	2-ядерный ARM A9 @ 1 GHz	ARMA9 @ 1 GHz	2-ядерный Intel Core i5 @ 1,9GHz
RAM	512 MB DDR3	1 GB DDR3	8 GB DDR3
Хранение	32 MB последовательная флэш	4 GB eMMC	64 GB SSD (опционно SATA)

Доступные ресурсы должны использоваться совместно с основными функциями маршрутизатора как сетевого агента, а также с тем, что пользователь разрабатывает как программное обеспечение для границ или туманов.

Граничному устройству необходимо управлять и обеспечивать безопасность развертываемых решений заказчика на краю, так как во многих ситуациях устройство будет очень далеко от программиста или администратора. Край необходимо учитывать в том же контексте, что и облачное программное обеспечение и сервисы. Маршрутизатор, предлагающий услуги граничных вычислений, предоставит API или интерфейс SDK для использования ресурсов на устройстве и развертывания программ.

Программное сетевое взаимодействие

Software-Defined Networking (SDN) – это метод развязки программного обеспечения и алгоритмов, определяющих плоскость управления сетью от базового оборудования, которое управляет плоскостью пересылки. Кроме того, **Network Function Virtualization (NFV)** определяется как предоставление сетевых функций, которые выполняются на аппаратных средствах поставщика. NFV описывает виртуализацию сетевых функций, обычно встречающихся на уровнях стека с четвертого по седьмой. Эти две парадигмы предоставляют отраслевые методы для построения, масштабирования и развертывания существенно сложной сетевой архитектуры очень гибким образом. Прежде всего,

это значительно снижает расходы предприятия на сетевую инфраструктуру, поскольку большинство сервисов могут работать в облаке.

Почему это важно для граничных устройств и как это вписывается в IoT? Мы освоили большую часть этой книги, в которой подробно описывалось перемещение данных с датчика в облако, однако, как принято считать, общая инфраструктура межсетевого взаимодействия будет масштабироваться до миллиарда дополнительных узлов в сети. Нужно только попросить ИТ-администратора разместить дополнительные миллионы оконечных точек в корпоративной сети, где узлы неоднородны, в отдаленных местах и некоторые движущиеся в транспортных средствах, чтобы понять влияние на общую сетевую инфраструктуру. Традиционные сети не масштабируются, и мы вынуждены рассматривать альтернативные способы построения крупных сетей с минимальными последствиями и стоимостью.

Архитектура SDN

В статье журнала «*Сетевое взаимодействие, основанное на программном обеспечении: всестороннее исследование*»[1] определяются четыре характеристики SDN:

- плоскость управления отделена от плоскости данных. Аппаратное обеспечение плоскости данных становится простым устройством пересылки пакетов;
- все решения по переадресации основаны на потоках, а не на адресе назначения. Поток – это набор пакетов, которые соответствуют критерию или фильтру. Все пакеты в потоке обрабатываются по тем же правилам пересылки и обслуживания. Программирование потоков дает легкость масштабирования и гибкость виртуальных коммутаторов, межсетевых экранов и промежуточного программного обеспечения;
- логика управления также известна как контроллер SDN. Эта версия программного обеспечения заменяет оборудование и способна работать на бюджетном оборудовании и облачных устройствах. Его цель – командовать и управлять упрощенными коммутирующими узлами. Досягаемость от абстракции контроллера SDN до узлов коммутации обеспечивается с помощью южного интерфейса;
- сетевое прикладное программное обеспечение может находиться на контроллере SDN и работать через северный интерфейс. Это программное обеспечение может взаимодействовать и манипулировать плоскостью данных с такими сервисами, как глубокая проверка пакетов, брандмауэры и балансировщики нагрузки.

Инфраструктура SDN похожа на традиционную сеть, поскольку использует аналогичное оборудование: коммутаторы, маршрутизаторы и средние устрой-

[1] *Крейц Д., Рамос Ф. М., Версиссимо П. Э., Ротенберг К. Э., Азодолмольский С., Улиг С.* Сетевое взаимодействие, основанное на программном обеспечении: всестороннее исследование // IEEE. Т. 103. № 1. С. 14–76. 2015. Янв.

ства. Основное различие, однако, состоит в том, что SDN использует готовые быстрые серверные вычислительные мощности без использования сложного и уникального встроенного управляющего оборудования. Эти серверные платформы обычно работают в облачных сетях, предоставляя сетевые сервисы программно, а не с помощью специализированных ASIC. Граничные маршрутизаторы, по сути, немыслимы без автономного управления. Архитектура SDN разделяет плоскость управления (логическое и функциональное управление) и плоскость данных (которая выполняет решения о путях данных и перенаправляет трафик). Плоскость данных состоит из маршрутизаторов и коммутаторов, которые связаны с контроллером SDN.

Все, что находится над оборудованием плоскости и пересылки данных, как правило, может находиться в облаке или на оборудовании персонального центра обработки данных (рис. 8.3).

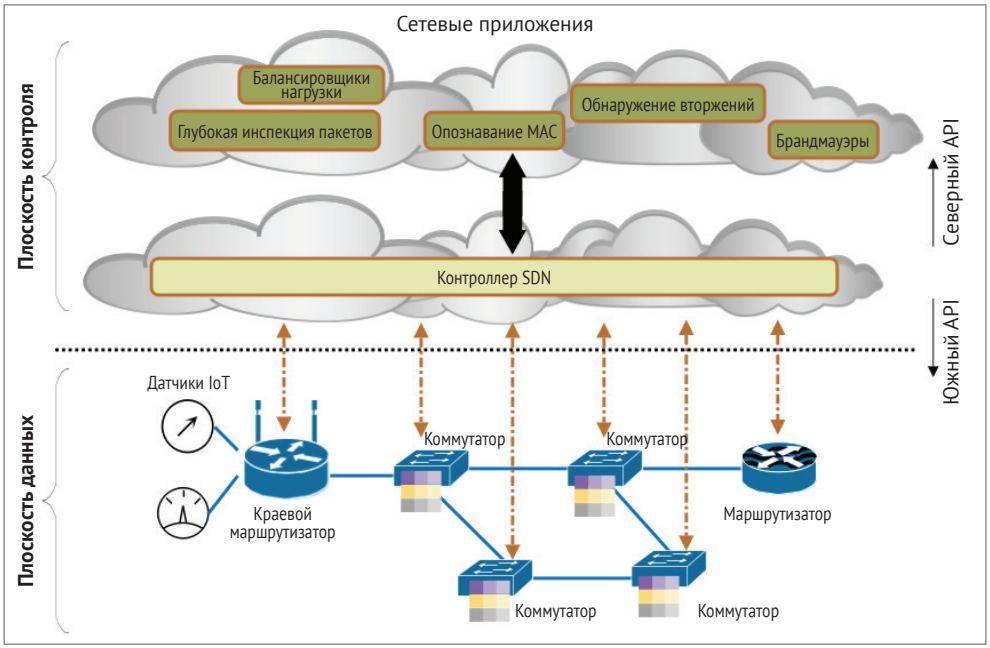


Рис. 8.3 ❖ Типичная абстракция архитектуры SDN

На иллюстрации показаны упрощенные узлы коммутации и пересылки, которые находятся на плоскости данных и направляют информацию по заданным путям, определенным абстрагированным контроллером SDN, который может располагаться в экземпляре облака. Контроллер SDN управляет плоскостью управления через южный интерфейс к узлам пересылки. Сетевые приложения могут находиться поверх контроллера SDN и управлять плоскостью данных с такими службами, как мониторинг угроз и обнаружение вторжений.

Эти службы обычно требуют специального и уникального оборудования для развертывания и управления клиентом.

Традиционное межсетевое взаимодействие

Типичная межсетевая архитектура будет использовать набор управляемых аппаратных/программных компонентов, которые предназначены для единственной цели и содержат встроенное программное обеспечение/решения. Часто они используют необычное оборудование и специализированные конструкции ASIC. Типичные функции включают в себя маршрутизацию, управляемые коммутаторы, брандмауэры, глубокую проверку пакетов и обнаружение вторжений, балансировщиков нагрузки и анализаторы данных. Часто такие специализированные устройства нуждаются в управлении клиентом и укомплектованы обученным сетевым ИТ-персоналом для их обслуживания и администрирования. Эти компоненты могут поставляться многими поставщиками и требуют существенно разных методов управления.

В этой конфигурации плоскость данных и плоскость управления унифицированы. Когда системе необходимо добавить или удалить какой-либо узел или настроить новый путь передачи данных, многие из выделенных систем необходимо обновить с новыми настройками VLAN, параметрами QoS, списками управления доступом, статическими маршрутами и правилами брандмауэра. Этим можно управлять при работе с несколькими тысячами оконечных точек. Однако, когда мы расширяемся до миллионов удаленных узлов, которые перемещаются и подключаются/отсоединяются, такая традиционная технология становится несостоятельной (рис. 8.4).

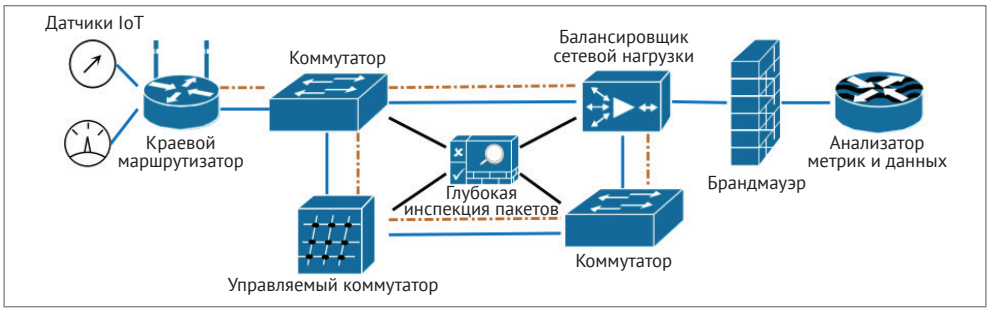


Рис. 8.4 ❖ Традиционные сетевые компоненты. В типичных сценариях межсетевого взаимодействия, предоставляющих такие сервисы, как безопасность, глубокий контроль пакетов, балансировка нагрузки и сбор показателей, требуются специальные аппаратные средства и системы управления под заказ. Это создает трудности в управлении и масштабировании для крупных инсталляций, удаленных устройств и подвижных систем, поскольку плоскости управления и данных соединены

Преимущества SDN

Для массового развертывания IoT следует учитывать модель сетевого взаимодействия SDN, особенно когда клиенту необходимо установить источник происхождения и обеспечить безопасность при широком развертывании узлов. Архитектор должен учитывать следующие ситуации при использовании SDN:

- серверы и центры обработки данных, с которыми должны взаимодействовать пограничные устройства IoT, могут находиться за тысячи километров;
- масштабы роста IoT от миллионов оконечных точек до миллиардов оконечных точек требуют соответствующих технологий масштабирования за пределами веерной модели современной интернет-инфраструктуры.

Три аспекта SDN делают его привлекательным для развертывания IoT:

- **цепочка обслуживания** – позволяет клиенту или поставщику продавать услуги alacarte (под заказ). Облачные сетевые сервисы, такие как брандмауэры, глубокая проверка пакетов, VPN, услуги аутентификации и брокеры политики, могут быть связаны и использоваться на основе подписки. Некоторым заказчикам может понадобиться полный набор функций, другие могут не выбрать ни одной или изменить свою конфигурацию в обычном ритме. Цепочка обслуживания обеспечивает значительную гибкость при развертывании;
- **динамическое управление нагрузкой** – SDN обладает гибкостью облачной архитектуры и по своему усмотрению может масштабировать ресурсы динамически в зависимости от нагрузки. Этот тип гибкости имеет решающее значение для IoT, поскольку архитекторам необходимо планировать емкость и масштаб, т. к. количество вещей растет экспоненциально. Только виртуальная сеть в облаке обеспечивает возможность масштабирования возможностей по мере необходимости. Примером этого может служить отслеживание людей в парках развлечений и других местах. Количество людей варьируется в зависимости от сезона, времени суток и погоды. Динамическая сеть может корректировать количество посетителей без каких-либо изменений в аппаратном обеспечении провайдера;
- **полоса пропускания по календарю**: это позволяет оператору разделять пропускную способность и использование данных в определенные времена и дни. Это относится к IoT, так как многие граничные датчики сообщают данные только периодически или в определенное время суток. Сложные алгоритмы разделения полосы пропускания могут быть построены с учетом сдвигов по времени.

Позже в этой книге, в главе 12, будут исследованы **программно-определенные параметры (SDP)** как еще один пример виртуализации сетевых функций и способ ее использования для создания микросегментов и изоляции устройств, что имеет решающее значение для безопасности IoT.

Заключение

Маршрутизаторы и шлюзы играют важную роль в разработке IoT. Функциональность, предоставляемая граничными маршрутизаторами, обеспечивает безопасность уровня предприятия, маршрутизацию, отказоустойчивость и качество обслуживания. Шлюз играет важную роль в переводе не-IP-сетей на IP-протоколы, что необходимо для подключения к интернету и облачным соединениям. Также важно понимать, что рост IoT до миллиардов узлов будет происходить с учетом требований низкой стоимости и ограниченной электроники. Такие функции, как корпоративная маршрутизация, туннелирование и VPN, требуют значительных аппаратных средств и возможностей обработки, и имеет смысл использовать маршрутизаторы и шлюзы для этой сервисной функции. Позже в этой книге мы также увидим, что граничные маршрутизаторы играют жизненно важную роль в граничной обработке и туманных вычислениях.

В следующей главе мы углубимся в протоколы на основе IoT, такие как MQTT и CoAP, с рабочими примерами. Эти протоколы являются легковесным языком IoT и чаще всего используют шлюзы и пограничные маршрутизаторы в качестве переводчиков.

Глава 9

IoT-протоколы передачи данных от граничного устройства в облако

До сих пор в этой книге мы рассматривали данные или события, сгенерированные на устройстве, работающем на краю сети. Мы обсудили различные телекоммуникационные среды и технологии для перемещения этих данных из WPAN, WLAN и WAN. Существует множество сложностей и тонкостей в построении и подключении сетей PAN, не основанных на IP, к сетям WAN на основе IP. Существуют также преобразования протоколов, которые необходимо понимать. Стандартные протоколы – это инструменты, которые связывают и инкапсулируют необработанные данные от датчика к чему-то значимому и форматируют их для облака. Одно из основных различий между системой IOT и системой M2M заключается в том, что M2M может связываться через WAN с выделенным сервером или системой без какого-либо инкапсулированного протокола. По определению IoT основан на связи между оконечными точками и службами, причем интернет является общей сетевой основой.

В этой главе также будут подробно описаны необходимые протоколы, распространенные в пространстве IoT, такие как **телеметрический транспорт очереди сообщений (MQTT)** и CoAP.

Протоколы

Естественный вопрос: почему существуют какие-либо протоколы за пределами HTTP для передачи данных по глобальной сети? HTTP обеспечивает значительные услуги и возможности для интернета уже более 20 лет, но был разработан и сконструирован для общего использования в моделях клиент/сервер. Устройства IoT могут быть очень стесненными, удаленными и ограниченными по полосе пропускания. Поэтому для управления множеством устройств в различных сетевых топологиях, таких как mesh-сети, необходимы более эффективные, безопасные и масштабируемые протоколы.

При передаче данных в интернет разработка переносится на фундаментальные уровни TCP/IP. Протоколы TCP и UDP являются очевидным и единственным выбором в передаче данных, TCP значительно сложнее в реализации, чем UDP (который является протоколом многоадресной рассылки). Однако UDP не обладает стабильностью и надежностью TCP, заставляя в некоторых разработках компенсировать это добавлением отказоустойчивости на уровнях приложений выше UDP.

Многие протоколы, перечисленные в этой главе, представляют собой реализации **промежуточного ПО, ориентированного на сообщения (MOM)**. Основная идея MOM заключается в том, что связь между двумя устройствами происходит с использованием распределенных очередей сообщений. MOM отправляет сообщения от одного приложения в пользовательском пространстве другому. Некоторые устройства производят данные для добавления в очередь, в то время как другие потребляют данные, находящиеся в очереди. Некоторые реализации требуют, чтобы центральным сервисом был брокер или посредник. В этом случае производители и потребители имеют публицистические и подписные связи с брокером. AMQP, MQTT и STOMP являются реализациями MOM; другие включают службы обмена сообщениями CORBA и Java. Реализация MOM с использованием очередей может использовать их для устойчивости в разработке. Данные могут сохраняться в очередях, даже если откажет сервер.

Альтернативой реализации MOM является RESTful. В модели RESTful сервер владеет состоянием ресурса, но состояние не передается в сообщении от клиента на сервер. RESTful использует HTTP-методы, такие как `GET`, `PUT`, `POST` и `DELETE` для размещения запросов по **универсальному идентификатору ресурса (URI)** (см. рис. 9.1). В этой архитектуре не требуется брокер или посредник. Поскольку они основаны на стеке HTTP, они пользуются большинством предлагаемых сервисов, таких как безопасность HTTPS. Проекты RESTful типичны для клиент-серверных архитектур. Клиенты инициируют доступ к ресурсам через синхронные шаблоны запроса-ответа.

Кроме того, клиенты несут ответственность за ошибки, даже если сервер падает. На рис. 9.1 показана MOM в сравнении с сервисом RESTful. Слева находится служба обмена сообщениями (на основе MQTT), использующая серединный брокерский сервер, издателей и подписчиков событий. Здесь многие клиенты могут быть как издателями, так и подписчиками, и информация может храниться или не храниться в очереди для быстрого восстановления. Справа находится проект RESTful, где архитектура построена на HTTP и использует HTTP-парадигмы для связи от клиента к серверу.

URI используется как идентификатор для передачи данных через интернет. Наиболее заметным URI является **универсальный локатор ресурса (URL)**, такой как http://www.iotforarchitects.net:8080/iot/?id="temperature". URI можно разбить на составные части, которые используются различными уровнями сетевого стека:

схема – `http://`;
основание – www.iotforarchitects.net;

порт – `8080`;
путь – `/iot`;
запрос – `?id="temperature"`.

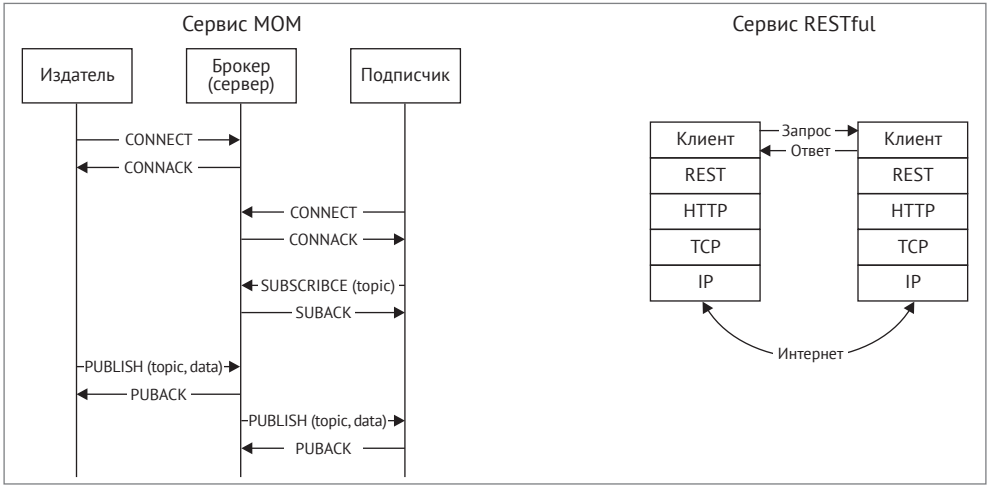


Рис. 9.1 ❖ Пример для сравнения MOM с реализацией RESTful

MQTT

Технология IBM Websphere Message Queue была впервые придумана в 1993 г. для решения проблем в независимых и неконкурентных распределенных системах для обеспечения защищенной связи. Производная от Web Sphere Message Queue была создана Энди Стэнфордом-Кларком и Арленом Ниппером в IBM в 1999 г. для решения конкретных проблем, связанных с подключением удаленных нефте- и газопроводов через спутниковую связь. Этот протокол стал известен как MQTT. Цели этого транспортного протокола, базирующегося на IP, следующие:

- он должен быть простым в реализации;
- обеспечивать форму качества обслуживания;
- быть очень легким и эффективным с точки зрения пропускной способности;
- быть платформенно-независимым;
- постоянное отслеживание сеанса;
- решение проблем безопасности.

MQTT обеспечивает выполнение всех этих требований. При рассмотрении протокола лучше всего пользоваться стандартным сайтом (mqtt.org), на котором приведена очень хорошо известная выборка из описания протокола:

«MQTT – это MQ Telemetry Transport, который представляет собой простой и легкий протокол обмена сообщениями, предназначенный для ограниченных устройств и сетей с низкой пропускной способностью, с высокой задержкой

или ненадежностью. Разработан на принципах минимизации пропускной способности сети и требований к ресурсам устройств, пытаясь в то же время обеспечить надежность и некоторую степень уверенности в доставке. Эти принципы также делают этот протокол идеальным для появления мира подключенных устройств типа «машина-машина» (M2M) или «интернет вещей», а также для мобильных приложений, где крайне важны пропускная способность и заряд батареи».

MQTT был внутренним и проприетарным протоколом для IBM в течение многих лет, пока не был выпущен в версии 3.1 в 2010 г. в качестве бесплатного продукта. В 2013 г. MQTT был стандартизирован и принят в консорциум OASIS. В 2014 г. OASIS опубликовал его публично как версию MQTT 3.1.1. MQTT также является стандартом ISO (ISO/IECPRF 20922).

Издание-подписка MQTT

В то время как архитектуры клиент-сервер многие годы являются основой для сервисов центров обработки, модели издание-подписка представляют собой альтернативу, которая полезна для использования IoT. Издание-подписка, также известная как **pub/sub**, является способом отделить клиента, передающего сообщение от другого клиента, получающего сообщение. В отличие от традиционной модели клиент-сервер, клиенты не осведомлены о каких-либо физических идентификаторах, наподобие IP-адреса или порта. MQTT – это архитектура pub/sub, но не очередь сообщений. Очереди сообщений по природе своей хранят сообщения, а MQTT – нет. В MQTT, если никто не подписывается (или не слушает) на тему, она просто игнорируется и теряется. Очереди сообщений также поддерживают топологию клиент-сервер, где один потребитель соединен с одним производителем.

В MQTT есть хранимые сообщения, которые будут рассмотрены позже. Хранимое сообщение представляет собой один экземпляр одного сообщения, которое сохранено.

Клиент, передающий сообщение, называется **издателем**; клиент, получающий сообщение, называется **подписчиком**. В центре находится брокер MQTT, который несет ответственность за соединение клиентов и фильтрацию данных. Такие фильтры обеспечивают:

- **фильтрация по темам** – по задумке, клиенты подписываются на темы и определенные ветви тем и не получают данных больше, чем хотят. Каждое опубликованное сообщение должно содержать тему, и брокер несет ответственность за повторную передачу этого сообщения подписчикам или игнорирование его;
- **фильтрация по содержимому** – брокеры имеют возможность проверять и фильтровать опубликованные данные. Таким образом, любые данные, которые не зашифрованы, могут управляться брокером до того, как сохранить их или передать другим клиентам;

- **фильтрация по типу** – клиент, прослушивающий поток данных, на которые он подписан, может также применять свои собственные фильтры. Входящие данные могут анализироваться, и в зависимости от этого поток данных обрабатывается далее или игнорируется.

В MQTT может быть много производителей и много потребителей, как показано на рис. 9.2.

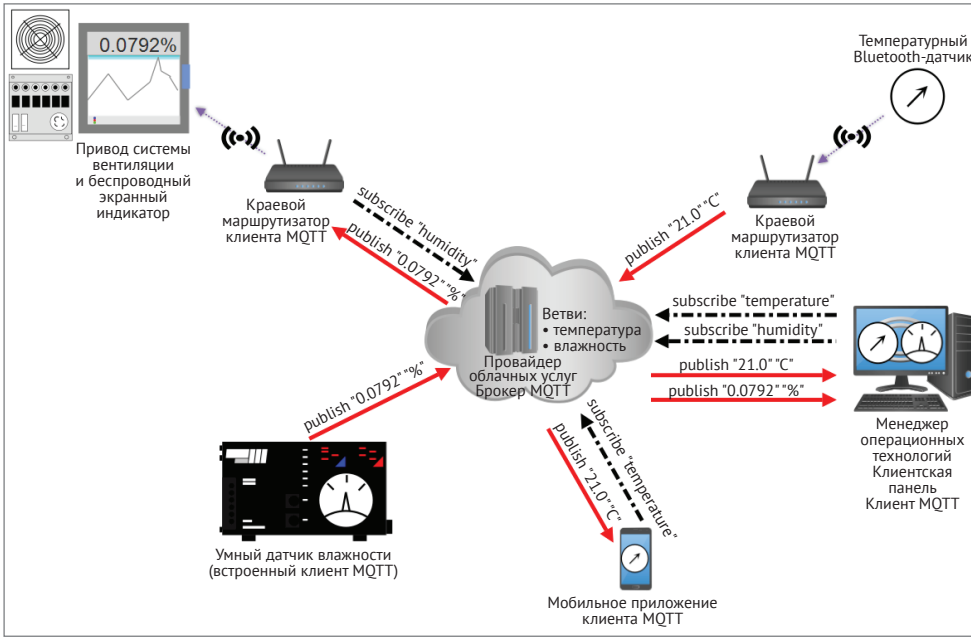


Рис. 9.2 ❖ Модель и топология издание-подписка MQTT. Клиенты работают на краю, публикуют и/или подписываются на темы, управляемые брокером MQTT. Здесь рассмотрены две темы: влажность и температура. Клиент может подписаться на несколько тем. На рисунке представлены умные датчики, обладающие достаточными ресурсами для управления собственным клиентом MQTT, а также пограничные маршрутизаторы, которые предоставляют клиентские услуги MQTT от имени датчиков или устройств, которые не поддерживают MQTT

Одно из предостережений модели вычислений издатель/подписчик заключается в том, что как издатель, так и подписчик должны знать тему ветви и формат данных перед началом передачи.

MQTT успешно отделяет издателей от потребителей. Поскольку брокер является руководящим органом между издателями и потребителями, нет необходимости напрямую идентифицировать издателя и потребителя на основе физических данных (таких как IP-адрес). Это очень полезно при развертывании IoT, поскольку физический идентификатор может быть неизвестным или

общим. MQTT и другие модели pub/sub также являются временно-независимыми. Это означает, что сообщение, опубликованное одним клиентом, подписчиком может прочитать и ответить на него в любое время. Абонент может находиться в месте с очень низким энергопотреблением / ограниченной полосой пропускания (например, Sigfox communication) и ответить на сообщение через несколько минут или часов. Из-за отсутствия физических и временных отношений модели pub/sub очень хорошо подходят для повышения производительности.

Управляемые облачно брокеры MQTT обычно могут поглощать миллионы сообщений в час и поддерживать десятки тысяч издателей.

MQTT не зависит от формата данных. Полезная нагрузка может содержать любой тип данных, поэтому и издатели, и подписчики должны понимать и согласовывать формат данных. Можно передавать текстовые сообщения, данные изображения, аудиоданные, зашифрованные данные, двоичные данные, объекты JSON или практически любую другую структуру в полезной нагрузке. Однако текстовые и двоичные данные JSON являются наиболее распространенными типами данных полезной нагрузки.

Максимально допустимый размер пакета в MQTT составляет 256 Мб, что позволяет получить чрезвычайно большую полезную нагрузку.

Обратите внимание, однако, что максимальный размер полезной нагрузки данных зависит от облака и брокера. Например, IBM Watson позволяет обрабатывать данные размером до 128 Кб, а Google поддерживает 256 Кб. С другой стороны, опубликованное сообщение может включать полезную нагрузку нулевой длины. Поле полезной нагрузки не является обязательным. Целесообразно сверить соответствие размеров полезной нагрузки с вашим облачным провайдером. Несоблюдение этого требования приведет к ошибкам и отключению от облачного брокера.

Детали архитектуры MQTT

Само название MQTT является неточным. В протоколе нет очереди сообщений. Хотя можно отправлять сообщения в очередь, это необязательно и часто не делается. MQTT основан на TCP и поэтому есть некоторая гарантия того, что пакет передается надежно.

MQTT также является *асимметричным протоколом*, тогда как HTTP – это *несимметричный протокол*. Скажем, узел A должен связываться с узлом B. Асимметричный протокол между A и B требует, чтобы протокол использовала только одна сторона (A), однако вся информация, необходимая для повторной сборки пакетов, должна содержаться в заголовке фрагментации, посланном A. В асимметричных системах есть один ведущий и один ведомый (FTP – классический пример). В симметричном протоколе он установлен как на A, так и на B. A или B могут принимать на себя роль ведущего или ведомого (основной пример – telnet). В MQTT роли разные, что имеет смысл в топологии датчиков/облаков.

MQTT может сохранять сообщение в брокере неограниченно долго. Этот режим работы управляется флагом при нормальной передаче сообщения. Со-

храненное на брокере сообщение отправляется любому клиенту, который подписывается на эту тематическую ветку MQTT. Сообщение немедленно отправляется этому новому клиенту. Это позволяет новому клиенту получить статус или сигнал из темы, на которую он недавно подписался, без ожидания. Как правило, клиент, подписывающийся на тему, может ожидать часы или даже дни, прежде чем клиент опубликует новые данные.

MQTT определяет дополнительный объект под названием **Последняя воля и завещание (LWT)**. LWT – это сообщение, которое указывает клиент во время фазы подключения. LWT содержит тему «Последняя воля», QoS и фактическое сообщение. Если клиент неправильно отключается от брокерского соединения (например, тайм-аут keep-alive, ошибка ввода-вывода или клиент закрывает сеанс без отключения), тогда брокер обязан транслировать сообщение LWT всем другим подписанным на эту тему клиентам.

Несмотря на то, что MQTT основан на TCP, соединения все еще могут обрываться, особенно в случае беспроводных датчиков. Устройство может потерять питание, потерять сильный сигнал или может быть просто полевая поломка, и сеанс перейдет в полуоткрытое состояние. Тогда сервер будет считать, что соединение по-прежнему надежно и ожидать данные. Чтобы выйти из этого полуоткрытого состояния, MQTT использует систему keep-alive. Используя эту систему, как брокер MQTT, так и клиент имеют гарантию того, что соединение остается работоспособным, даже если в течение некоторого времени не было передачи. Клиент отправляет пакет PINGREQ брокеру, который, в свою очередь, подтверждает сообщение с помощью PINGRESP. Таймер установлен на стороне клиента и брокера. Если сообщение не было передано ни кем из них в течение заданного промежутка времени, должен быть отправлен пакет keep-alive. Как PINGREQ, так и сообщение, сбросят таймер keep-alive.

В случае, если keep-alive не получен и время таймера истекает, брокер закроет соединение и отправит LWT-пакет всем клиентам. Клиент может в какой-то момент позже попытаться снова подключиться. В этом случае брокер закрывает полуоткрытое соединение и открывает новое соединение с клиентом.

Максимальное время ожидания – 18 часов, 12 минут и 15 с. Установка внутреннего состояния keep-alive на 0 приведет к отключению функции keep-alive. Таймер управляется клиентом и может динамически изменяться для отражения режимов сна или изменения уровня сигнала.

В то время как keep-alive помогает с нарушенными соединениями, повторное установление всех подписок клиента и параметров QoS может привести к ненужным накладным расходам на подключенном соединении. Чтобы уменьшить эти дополнительные расходы, MQTT позволяет поддерживать постоянные соединения. Постоянное соединение сохраняет на стороне брокера следующее:

- все подписки клиента;
- все сообщения QoS, которые не были подтверждены клиентом;
- все новые сообщения QoS, пропущенные клиентом.

Параметр `client_id` ссылается на эту информацию для уникальной идентификации клиентов. Клиент может запрашивать постоянное соединение, однако брокер может отклонить запрос и принудительно перезапустить чистый сеанс. При соединении брокером используется флажок `cleanSession` для разрешения или запрета постоянных соединений. Клиент может определить, сохранялось ли постоянное соединение с помощью сообщения CONNACK.

Постоянные сеансы должны использоваться для клиентов, которые должны получать все сообщения, даже когда нет связи. Они не должны использоваться в ситуациях, когда клиент только публикует (записывает) данные в темы.

В MQTT есть три уровня качества обслуживания:

- **QoS-0 (незаверенная передача)** – это минимальный уровень QoS. Это аналогично модели «сжечь и забыть», подробно описанной в некоторых беспроводных протоколах. Это самый эффективный процесс доставки без подтверждения получателем сообщения и без повторной передачи сообщения отправителем;
- **QoS-1 (гарантированная передача)** – этот режим гарантирует доставку сообщения хотя бы один раз получателю. Сообщение может быть доставлено несколько раз, и получатель отправит обратно подтверждение с ответом PUBACK;
- **QoS-2 (гарантированный сервис для приложений)** – это самый высокий уровень QoS, который убеждается в доставке и информирует отправителя и получателя, что сообщение было передано правильно. Этот режим генерирует больше трафика из-за многошагового рукопожатия между отправителем и получателем. Если получатель получает сообщение с уровнем обслуживания QoS-2, он ответит отправителю сообщением PUBREC. Это подтверждает сообщение, и отправитель ответит сообщением PUBREL. PUBREL позволяет получателю безопасно отбросить любые повторные передачи этого сообщения. Затем PUBREL подтверждается получателем с помощью PUBCOMP. Пока сообщение PUBCOMP не будет отправлено, приемник будет кэшировать исходное сообщение для обеспечения безопасности.

QoS в MQTT определяется и контролируется отправителем, и у каждого отправителя может быть своя политика.

Типичные случаи использования:

- **QoS-0** следует использовать, когда сообщение не требуется сохранять в очереди. QoS-0 лучше всего подходит для проводных соединений или когда система сильно ограничена в пропускной способности;
- **Qos-1** следует использовать по умолчанию. QoS1 намного быстрее, чем QoS2, и значительно снижает стоимость передачи;
- **QoS-2** – для критически важных приложений. Кроме того, для случаев, когда повторная передача дублированного сообщения может привести к ошибкам.

Рекламирование и обнаружение шлюза

Поскольку топология MQTT-SN несколько сложнее, чем MQTT, для установления маршрутов используется процесс обнаружения через несколько шлюзов и узлов для пересылки. Шлюзы, объединяющие топологию MQTT-SN, начинаются с привязки к брокеру MQTT. Затем он может выдать пакет ADVERTISE подключенным клиентам или другим шлюзам. В сети могут сосуществовать несколько шлюзов, но клиент может подключаться только к одному шлюзу. Клиент несет ответственность за сохранение списка активных шлюзов и их сетевых адресов. Этот список построен из различных широковещательных сообщений ADVERTISEMENT и GWINFO.

Из-за новых типов шлюзов и топологий в MQTT-SN доступно несколько новых сообщений для помощи в обнаружении и рекламировании:

- **ADVERTISE** – широковещательная рассылка с шлюза для рекламирования своего присутствия;
- **SEARCHGW** – широковещательная передача клиентом при поиске определенного шлюза. Часть сообщения представляет собой параметр `radius`, в котором указано, сколько переходов должно выполняться в поисковой сети по запросу SEARCHGW. Например, значение 1 указывает на один прыжок в очень плотной сети, где каждый клиент доступен за один прыжок;
- **GWINFO** – это ответ шлюзов после получения сообщения SEARCHGW. Он содержит идентификатор и адрес шлюза, который передается только при отправке клиентом SEARCHGW.

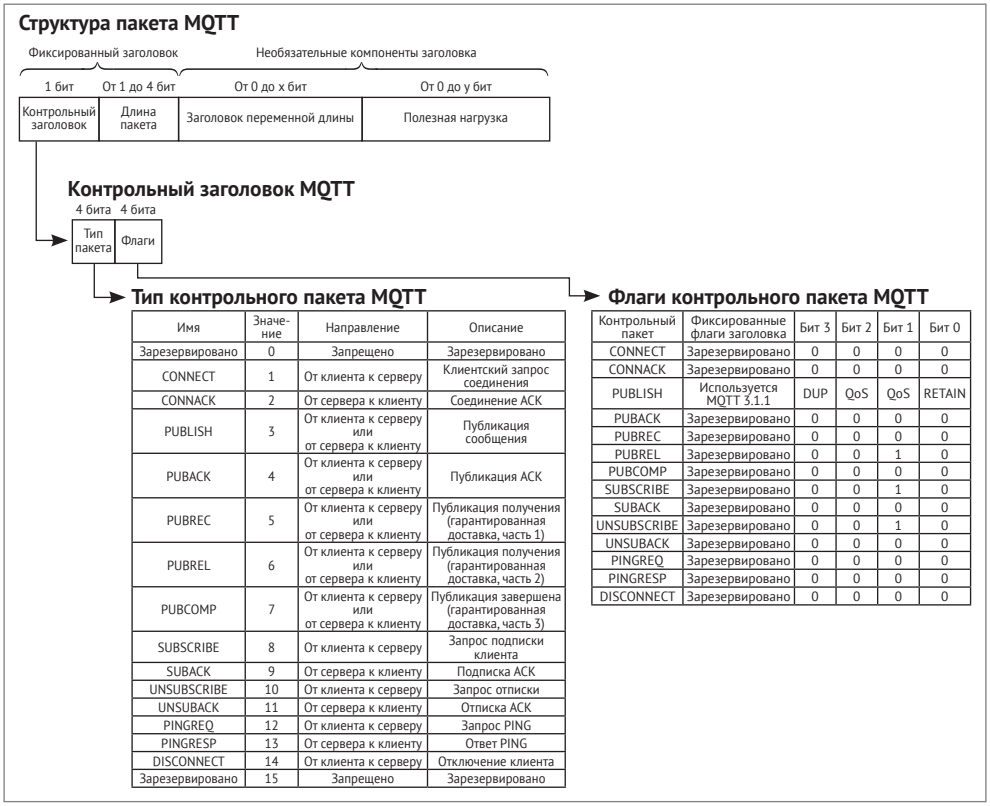
Структура пакета MQTT

Пакет MQTT находится сверху уровня TCP сетевого стека в модели OSI. Пакет состоит из 2-байтового фиксированного заголовка, который всегда должен присутствовать, заголовка с переменным размером (необязательно) и заканчивается полезной нагрузкой (тоже необязательно) (рис. 9.3).

Форматы соединений MQTT

Соединение с использованием MQTT начинается с того, что клиент отправляет сообщение CONNECT брокеру. Только клиент может инициировать сеанс, и ни один клиент не может напрямую связаться с другим клиентом. В ответ на сообщение CONNECT брокер всегда будет отсылать CONNACK и код статуса. После установления соединение остается открытым. В табл. 9.1 приведены сообщения и форматы MQTT.

- **Формат CONNECT (клиент к серверу)**: типичное сообщение CONNECT будет содержать данные (для запуска сеанса требуется только `clientID`), показанные в табл. 9.1.



Имя	Значение	Направление	Описание
Зарезервировано	0	Запрещено	Зарезервировано
CONNECT	1	От клиента к серверу	Клиентский запрос соединения
CONNACK	2	От сервера к клиенту	Соединение ACK
PUBLISH	3	От клиента к серверу или от сервера к клиенту	Публикация сообщения
PUBACK	4	От клиента к серверу или от сервера к клиенту	Публикация ACK
PUBREC	5	От клиента к серверу или от сервера к клиенту	Публикация получения (гарантированная доставка, часть 1)
PUBREL	6	От клиента к серверу или от сервера к клиенту	Публикация получения (гарантированная доставка, часть 2)
PUBCOMP	7	От клиента к серверу или от сервера к клиенту	Публикация завершена (гарантированная доставка, часть 3)
SUBSCRIBE	8	От клиента к серверу	Запрос подписки клиента
SUBACK	9	От сервера к клиенту	Подписка ACK
UNSUBSCRIBE	10	От клиента к серверу	Запрос отписки
UNSUBACK	11	От сервера к клиенту	Отписка ACK
PINGREQ	12	От клиента к серверу	Запрос PING
PINGRESP	13	От сервера к клиенту	Ответ PING
DISCONNECT	14	От клиента к серверу	Отключение клиента
Зарезервировано	15	Запрещено	Зарезервировано

Контрольный пакет	Фиксированные флаги заголовка	Бит 3	Бит 2	Бит 1	Бит 0
CONNECT	Зарезервировано	0	0	0	0
CONNACK	Зарезервировано	0	0	0	0
PUBLISH	Используется MQTT 3.1.1	DUP	QoS	QoS	RETAIN
PUBACK	Зарезервировано	0	0	0	0
PUBREC	Зарезервировано	0	0	0	0
PUBREL	Зарезервировано	0	0	1	0
PUBCOMP	Зарезервировано	0	0	0	0
SUBSCRIBE	Зарезервировано	0	0	1	0
SUBACK	Зарезервировано	0	0	0	0
UNSUBSCRIBE	Зарезервировано	0	0	1	0
UNSUBACK	Зарезервировано	0	0	0	0
PINGREQ	Зарезервировано	0	0	0	0
PINGRESP	Зарезервировано	0	0	0	0
DISCONNECT	Зарезервировано	0	0	0	0

Рис. 9.3 ❖ Общая структура пакетов MQTT

Таблица 9.1. Сообщения и форматы MQTT

Поле	Необходимость	Описание
`clientID`	Требуется	Идентифицирует клиента на сервере. Каждый клиент имеет уникальный идентификатор клиента. Он может составлять от 1 до 23 байтов UTF-8
`cleanSession`	Необязательно	0: сервер должен возобновить связь с клиентом. Клиент и сервер должны сохранить состояние сеанса после отключения. 1: Клиент и сервер должны отменить предыдущий сеанс и начать новый
`username`	Необязательно	Имя, используемое сервером для аутентификации
`password`	Необязательно	Двоичный пароль длиной от 0 до 65536 байтов с префиксом 2 байта
`lastWillTopic`	Необязательно	Тема ветви для публикации сообщения
`lastWillQos`	Необязательно	2 бита с указанием уровня QoS при публикации сообщения последней воли

Таблица 9.1 (окончание)

Поле	Необходимость	Описание
`lastWillMessage`	Необязательно	Определяет полезную нагрузку сообщения последней воли
`lastWillRetain`	Необязательно	Указывает, сохраняется ли последняя воля после публикации
`keepAlive`	Необязательно	Интервал времени в секундах. Клиент отвечает за отправку сообщения или пакета PINGREQ до истечения таймера `keepAlive`. Сервер отключится от сети в через 1,5 указанного интервала в режиме keep-alive. Значение 0 (0) отключит механизм `keepAlive`

- **Коды возврата CONNECT (от сервера к клиенту)**: брокер будет отвечать на сообщение CONNECT с кодом ответа. Дизайнер должен знать, что не все соединения могут быть одобрены брокером. Коды ответа показаны в табл. 9.2.

Таблица 9.2. Коды возврата CONNECT (от сервера к клиенту)

Код возврата	Описание
`0`	Успешное соединение
`1`	В соединении отказано: неприемлемая версия протокола MQTT
`2`	В соединении отказано: идентификатор клиента – это правильный UTF-8, но не разрешенный сервером
`3`	В соединении отказано: сервер недоступен
`4`	В соединении отказано: неправильное имя пользователя или пароль
`5`	В соединении отказано: клиент не авторизован для соединения

- **Формат PUBLISH (клиент к серверу)**: на данный момент клиент может публиковать данные в ветке темы. Каждое сообщение содержит тему (табл. 9.3).

Таблица 9.3. Формат PUBLISH (клиент к серверу)

Поле	Необходимость	Описание
`packetID`	Требуется	Уникально идентифицирует пакет в переменном заголовке. В зоне ответственности клиентской библиотеки. Всегда установлен на ноль (0) для QoS-0
`topicName`	Требуется	Темы ветви для публикации (например, США/Висконсин/Милуоки/температура)
`qos`	Требуется	QoS уровня 0, 1 или 2
`retainFlag`	Требуется	Имя, используемое сервером для аутентификации
`payload`	Необязательно	Полезная нагрузка в любом формате
`dupFlag`	Требуется	Сообщение является дубликатом и послано повторно

- **Формат SUBSCRIBE (от клиента к серверу)**: полезная нагрузка пакета подписки включает в себя по меньшей мере одну пару кодированных UTF-8 `topicID` и уровни `QoS`. В этой полезной нагрузке может быть указано несколько `topicID`, чтобы избавить клиента от нескольких передач (табл. 9.4).

Таблица 9.4. Формат SUBSCRIBE (от клиента к серверу)

Поле	Необходимость	Описание
`packetID`	Требуется	Уникально идентифицирует пакет в переменном заголовке. Ответственность клиентской библиотеки
`topic_1`	Требуется	Подписная тема ветки.
`qos_1`	Требуется	Уровень обслуживания QoS-сообщений, опубликованных в `topic_1`
`topic_2`	Необязательно	Имя, используемое сервером для аутентификации
`qos2`	Необязательно	Уровень обслуживания QoS-сообщений, опубликованных в `topic_2`

Подстановочные знаки могут использоваться для подписки на несколько тем в одном сообщении. Для этих примеров темой будет полный путь "`{страна}/{штаты}/{города}/{температура, влажность}`".

- **+ подстановочный шаблон одного уровня** – заменяет один уровень в имени строки темы. Например, US/+/Milwaukee заменит уровень штата всеми 50 штатами, от Аляски до Вайоминга;
- *** многоуровневый шаблон** – заменяет несколько уровней, а не один. Он всегда является последним символом в названии темы. Например, US/Wisconsin/* будут соответствовать всем городам Висконсина: Милуоки, Мэдисон, Глендейл, Уайтфиш-Бей, Брукфилд и так далее;
- **$ специальные темы** – это специальный статистический режим для брокеров MQTT. Клиенты не могут публиковать в темах $. На данный момент официального стандарта для использования нет. Одна из моделей использует $SYS следующим образом: $SYS/broker/clients/connected.

Сервер MQTT должен (но это не требуется спецификацией явно) поддерживать подстановочные знаки в названии темы. Если он не поддерживает подстановочные знаки, сервер должен их отклонить. Установка идентификатора `packetID` является ответственностью клиентской библиотеки MQTT.

В спецификации MQTT содержится несколько других сообщений. Более подробную информацию об API программирования MQTT можно найти в OASIS MQTT Version 3.1.1 Standard, docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf, 2014 г.

Рабочий пример MQTT

Для рабочего примера **Google Cloud Platform (GCP)** будет использоваться как приемник MQTT и излучатель в облако. Большинство облачных сервисов MQTT следуют аналогичной модели, поэтому этот фреймворк можно использовать в качестве ссылки. Мы будем использовать инструменты с открытым исходным кодом, чтобы запустить клиент MQTT и простой пример Python для публикации строки «здравствуй, мир» в теме ветки. Для начала использования GCP необходимы предварительные шаги. Перед тем, как продолжить, необходимо обеспечить безопасность аккаунтов и платежных систем Google. Пожалуйста, ознакомьтесь с этими инструкциями до начала работы с Google IoT Core cloud.google.com/iot/docs/how-tos/getting-started.

Продолжайте работать в GCP, чтобы создать устройство, включить Google API, создать ветвь тем и добавить участника в издателя pub/sub.

Google уникален тем, что для шифрования всех пакетов данных с использованием **JSON Web Tokens (JWT)** и агента сертификата требуется сильное шифрование (TLS) поверх MQTT. Каждое устройство создаст пару открытых/закрытых ключей. Google гарантирует, что каждое устройство имеет уникальный идентификатор и ключ; в случае взлома это затронет только один узел и укажет на общую поверхность атаки.

Брокер MQTT начинает с импорта нескольких библиотек.

Библиотека **paho.mqtt.client** Python является проектом, спонсируемым Eclipse, и представляет собой ссылку на исходный проект IBM MQTT. Paho также является основным продуктом рабочей группы Eclipse M2M Industry Working Group. Существуют и другие варианты брокеров сообщений MQTT, такие как проект Eclipse Mosquitto и RabbitMQ:

```
#Простой пример MQTT Client для Google Cloud Platform
import datetime
import os
import time
import paho.mqtt.client as mqtt
import jwt

project_id = 'name_of_your_project'
cloud_region = 'us-central1'
registry_id = 'name_of_your_registry'
device_id = 'name_of_your_device'
algorithm = 'RS256'
mqtt_hostname = 'mqtt.googleapis.com'
mqtt_port = 8883
ca_certs_name = 'roots.pem'
private_key_file = '/Users/joeuser/mqtt/rsa_private.pem'
```

Следующим шагом будет аутентификация с Google с помощью ключа. Здесь мы используем объект JWT для хранения сертификата:

```
#Google требует аутентификацию на основе сертификатов с использованием JSON Web Tokens
(JWT) на каждом устройстве.
#Ограничение области атаки
def create_jwt(project_id, private_key_file, algorithm):
    token = {
            # Время выдачи токена
            'iat': datetime.datetime.utcnow(),
            # Истечение времени токена.
            'exp': datetime.datetime.utcnow() +datetime.timedelta(minutes=60),
            # Поле Audience = project_id
            'aud': project_id
    }

    # Чтение файла закрытого ключа.
    with open(private_key_file, 'r') as f:
        private_key = f.read()
    return jwt.encode(token, private_key, algorithm=algorithm)
```

Мы определяем несколько обратных вызовов, используя библиотеку MQTT, такие как ошибки, соединения, отключения и публикации:

```
#Типичные обратные вызовы MQTT
def error_str(rc):
    return '{}: {}'.format(rc, mqtt.error_string(rc))

def on_connect(unused_client, unused_userdata, unused_flags, rc):
    print('on_connect', error_str(rc))

def on_disconnect(unused_client, unused_userdata, rc):
    print('on_disconnect', error_str(rc))

def on_publish(unused_client, unused_userdata, unused_mid):
    print('on_publish')
```

Далее следует основная структура клиента MQTT. Во-первых, мы регистрируем клиента, как предписывает Google. Google IoT требует определения проекта, региона, идентификатора регистрации и идентификатора устройства. Мы также пропустим имя пользователя и используем поле пароля с помощью метода `create_jwt`. Также мы включаем SSL-шифрование в MQTT – многие провайдеры облака MQTT требуют выполнение этого условия. После подключения к облачному MQTT-серверу Google основной цикл программы публикует простую строку helloworld для той ветви, на которую подписаны. Также следует отметить уровень QoS, который установлен в публикуемом сообщении.

Если требуется параметр, но он явно не установлен в программе, клиентская библиотека обязана использовать значения по умолчанию (например, флаги RETAIN и DUP используются как значения по умолчанию во время сообщения PUBLISH):

```
def main():
    client = mqtt.Client(
            client_id=('projects/{}/locations/{}/registries/{}/devices/{}'
                       .format(
                               project_id,
                               cloud_region,
                               registry_id,
                               device_id))) #Google требует такой формат явно

    client.username_pw_set(
            username='unused', #Google игнорирует имя пользователя.
            password=create_jwt( #Google требует JWT для авторизации
                    project_id, private_key_file, algorithm))

    # Активация поддержки SSL/TLS.
    client.tls_set(ca_certs=ca_certs_name)

    #функции обратного вызова в этом примере не используются:
    client.on_connect = on_connect
    client.on_publish = on_publish
    client.on_disconnect = on_disconnect

    # Подключение к Google pub/sub
    client.connect(mqtt_hostname, mqtt_port)
```

```
    # Цикл
    client.loop_start()

    # Публикация темы событий или состояний согласно флагу.
    sub_topic = 'events'
    mqtt_topic = '/devices/{}/{}'.format(device_id, sub_topic)

    # Публикация сообщений num_messages на мосту MQTT раз в секунду.

    for i in range(1,10):
        payload = 'Hello World!: {}'.format(i)
        print('Publishing message\'{}\''.format(payload))
        client.publish(mqtt_topic, payload, qos=1)
        time.sleep(1)

if __name__ == '__main__':
    main()
```

MQTT-SN

Производная для сетей датчиков от MQTT называется MQTT-SN (иногда звучит как MQTT-S). Она придерживается той же философии MQTT, что и легкий протокол для периферийных устройств, но сконструирована специально для нюансов беспроводной локальной сети, типичной для сенсорных сред. Это означает поддержку каналов с низкой пропускной способностью, отслеживание отказа канала, короткие сообщения и аппаратное обеспечение с ограниченными ресурсами. MQTT-SN, по сути, настолько прозрачен, что может успешно работать поверх BLE и Zigbee.

Для MQTT-SN не требуется стек TCP/IP. Его можно использовать по последовательному соединению (предпочтительный вариант), где простой протокол связи (чтобы различать разные устройства на линии) и накладные расходы очень малы. В качестве альтернативы он может использоваться протокол UDP, который требует меньше ресурсов, чем TCP.

Архитектура и топология MQTT-SN

В топологии MQTT-SN имеется четыре основных компонента (рис. 9.4):

- **шлюзы** – в MQTT-SN шлюз несет ответственность за преобразование протокола из MQTT-SN в MQTT и наоборот (хотя возможны и другие переводы). Шлюзы также могут быть агрегирующими или прозрачными (рассматривается далее в этой главе);
- **форвардеры** – маршрут между датчиком и шлюзом MQTT-SN может проходить по множеству путей и пересекать несколько маршрутизаторов. Узлы между исходным клиентом и шлюзом MQTT-SN называются пересылками, они просто повторно инкапсулируют кадры MQTT-SN в новые и неизменные кадры MQTT-SN, которые отправляются в пункт назначения до тех пор, пока они не достигнут правильного шлюза MQTT-SN для преобразования протокола;

- **клиенты** – клиенты ведут себя так же, как в MQTT, могут подписываться и публиковать данные;
- **брокеры** – брокеры ведут себя так же, как в MQTT.

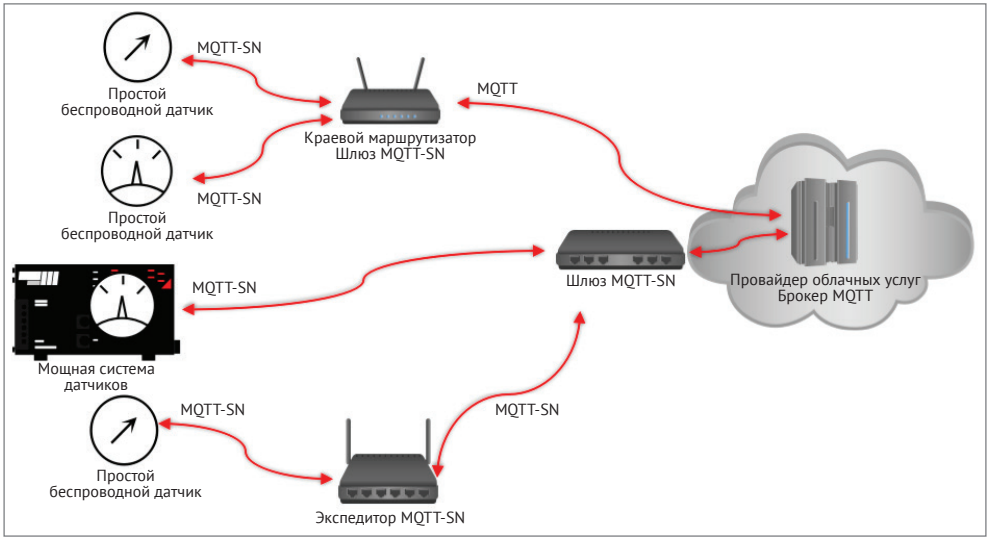


Рис. 9.4 ❖ Топология MQTT-SN. Беспроводные датчики связываются либо с шлюзами MQTT-SN, которые переводят MQTT-SN в MQTT, в другие протокольные формы, либо в форвардеров, которые просто инкапсулируют кадры MQTT-SN, принятые в сообщениях MQTT-SN, перенаправленные на шлюз

Прозрачные и собирающие шлюзы

В MQTT-SN шлюзы могут играть две разные роли (рис. 9.5). Во-первых, прозрачный шлюз будет управлять многими независимыми потоками MQTT-SN с сенсорных устройств и преобразовывать каждый поток в сообщение MQTT. Агрегационный/собирающий шлюз объединит несколько потоков MQTT-SN в меньшее количество потоков MQTT, отправленных к облачному брокеру MQTT. Агрегационный шлюз сложнее в создании, но он уменьшит количество передаваемых служебных данных и количество одновременных подключений, открытых на сервере. Для создания топологии агрегирующего шлюза клиенты должны публиковать в той же теме или подписываться на одну и ту же тему.

Различия между MQTT и MQTT-SN

Основные отличия между MQTT-SN и MQTT заключаются в следующем:

- в MQTT-SN есть три сообщения CONNECT, а в MQTT – одно. Дополнительные два используются для передачи темы «Will» и сообщения «Will»;
- MQTT-SN может работать через упрощенную среду и UDP;

- имена тем заменяются короткими двухбайтовыми сообщениями с идентификатором темы. Это необходимо для лучшего использования пропускной способности в беспроводных сетях;
- предварительно определенные идентификаторы тем и короткие имена тем могут использоваться без какой-либо регистрации. Чтобы использовать эту функцию, клиент и сервер должны использовать один и тот же идентификатор темы. Короткие названия достаточно коротки, чтобы быть встроенными в сообщение PUBLISH;
- предложена процедура обнаружения, помогающая клиентам и позволяющая им находить сетевые адреса серверов и шлюзов. В сети могут существовать несколько шлюзов, которые могут использоваться для обмена сообщениями с клиентами;
- функция `CleanSession` получает свойство Will. Клиентская подписка может быть сохранена, но теперь данные Will также сохраняются;
- в MQTT-SN используется пересмотренная процедура `keepAlive`. Это делается для поддержки спящих клиентов, в которых все предназначенные для них сообщения буферизуются сервером или пограничным маршрутизатором и передаются при пробуждении.

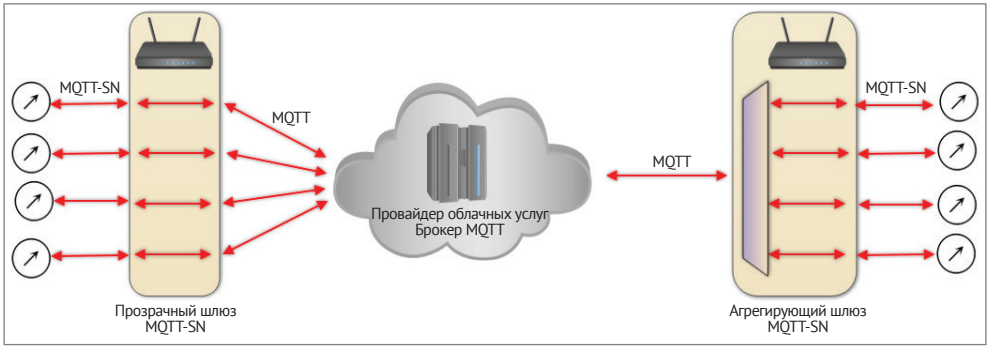


Рис. 9.5 ❖ Конфигурации шлюзов MQTT-SN. Прозрачный шлюз просто выполняет преобразование протокола для каждого входящего потока MQTT-SN и имеет взаимно-однозначное отношение между соединениями MQTT-SN и MQTT-соединениями с брокером. Однако агрегирующий шлюз объединит несколько потоков MQTT-SN в одно соединение MQTT с сервером

Ограниченный прикладной протокол

Ограниченный протокол приложений (CoAP) является разработкой IETF (RFC7228). Рабочая группа IETF **Constreined RESTful Environments (CoRE)** создала первый проект протокола в июне 2014 г., но еще несколько лет работала над его созданием. Он специально предназначен в качестве протокола связи для устройств с ограниченными возможностями. Основной протокол

теперь основан на RFC7252. Протокол уникален, поскольку он впервые разработан для обмена данными типа **машина-машина (M2M)** между граничными узлами. Он также поддерживает работу с HTTP с использованием прокси. Это преобразование HTTP является встроенным средством для получения данных через интернет.

CoAP превосходно обеспечивает аналогичную и легкую структуру адресации ресурсов, знакомую всем, у кого есть опыт использования интернета, но с ограниченными ресурсами и требованиями к пропускной способности. Исследование, проведенное Colitti et. al, продемонстрировало эффективность CoAP по сравнению со стандартным HTTP (*Colitti, Walter & Steenhaut, Kris & De, Niccolò. (2017*). Интеграция беспроводных сенсорных сетей с интернетом). CoAP обеспечивает аналогичную функциональность со значительно меньшими затратами и потребностями в мощности.

Кроме того, некоторые реализации CoAP выполняются на x64 лучше, чем эквиваленты HTTP на аналогичном оборудовании (табл. 9.5).

Таблица 9.5. Различия протоколов CoAP и HTTP

	Байты в транзакции	**Потребляемая мощность**	**Время жизни батареи**
CoAP	154	0,744 mW	151 день
HTTP	1451	1,333 mW	84 дня

Детали архитектуры CoAP

Концепция CoAP основана на имитации и замене тяжелых элементов HTTP и использования легкого эквивалента для IoT. Это не замена HTTP, поскольку в ней отсутствуют некоторые качества; HTTP требует более мощных и ориентированных на сервисы систем. Возможности CoAP можно суммировать следующим образом:

- похожа на HTTP;
- протоколы без установления соединения;
- безопасность с помощью DTLS, а не TLS, как при нормальной передаче HTTP;
- асинхронный обмен сообщениями;
- легкий дизайн и низкие требования к ресурсам, низкие накладные расходы в плане заголовка;
- поддержка URI и типов содержимого;
- построен на UDP вместо TCP/UDP, как для обычного HTTP-сеанса;
- HTTP-сопоставление без сохранения состояния, позволяющее прокси-серверу подключаться к HTTP-сеансам.

CoAP имеет два основных уровня (рис. 9.6):

- **уровень запроса/ответа** – отвечает за отправку и получение запросов на основе RESTful. Запросы REST включены в сообщения CON или NON. Ответ REST включен в соответствующее сообщение ACK;

- **уровень транзакций** – на этом уровне поддерживается обмен сообщениями между оконечными точками, используя один из четырех основных типов сообщений. Уровень транзакции также поддерживает управление многоадресной рассылкой и перегрузкой.

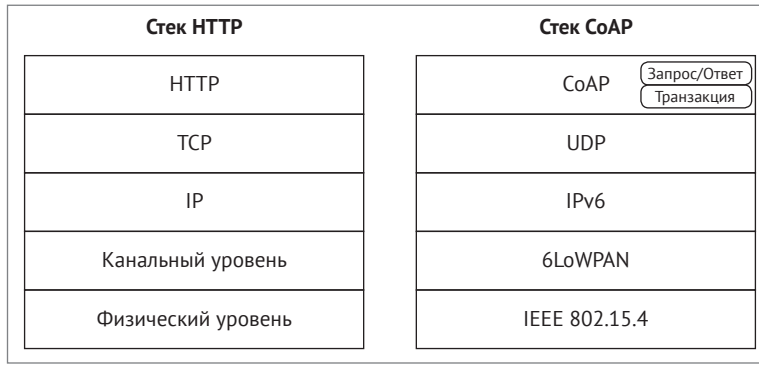


Рис. 9.6 ❖ Стек HTTP по сравнению с CoAP

Контекст, синтаксис и использование CoAP аналогично HTTP. Адресация в CoAP также в стиле HTTP. Адрес – это структура URI. Как и для URI в HTTP, пользователь должен знать адрес заранее, чтобы получить доступ к ресурсу. На самом верхнем уровне CoAP использует такие запросы, как `GET`, `PUT`, `POST` и `DELETE`, как в HTTP. Аналогично, коды ответов имитируют HTTP, например:

- `2.01` – создано;
- `2.02` – удалено;
- `2.04` – изменено;
- `2.05` – содержимое;
- `4.04` – не найдено (ресурс);
- `4.05` – метод не разрешен.

Типичный URI в CoAP имеет следующую форму:

```
coap://хост[:порт]/[путь][?запрос]
```

Система CoAP состоит из семи основных участников:

- **оконечные точки** – это источники и адресаты сообщения CoAP. Конкретное определение оконечной точки зависит от используемого транспорта;
- **прокси** – оконечная точка CoAP, которой клиенты CoAP поручают выполнять запросы от своего имени. Снижение нагрузки на сеть, доступ к спящим узлам и обеспечение некоторого уровня безопасности – вот некоторые из функций прокси. Прокси могут быть явно выбраны клиентом (прямое проксирование) или могут использоваться как серверы in-situ (обратное проксирование). В качестве альтернативы прокси может

преобразовать один запрос CoAP в другой запрос CoAP или даже перевести на другой протокол (перекрестное проксирование). Общей ситуацией является пограничный маршрутизатор, проксирующий из сети CoAP до сервисов HTTP для облачных интернет-соединений;

- **клиент**: инициатор запроса. Оконечная точка назначения ответа;
- **сервер**: оконечная точка назначения запроса. Создатель ответа;
- **посредник**: клиент, выступающий в качестве и сервера и клиента по отношению к исходному серверу. Прокси является посредником;
- **сервер происхождения**: сервер, на котором находится данный ресурс;
- **наблюдатели**: клиент наблюдателя может зарегистрировать себя с использованием измененного сообщения `GET`. Затем наблюдатель подключается к ресурсу, и, если состояние этого ресурса изменяется, сервер отправляет уведомление наблюдателю.

Наблюдатели уникальны в CoAP и позволяют устройству следить за изменениями в конкретном ресурсе. По сути, это похоже на модель подписки MQTT, где узел подписывается на событие.

Ниже приведен пример архитектуры CoAP. Будучи легкой HTTP-системой, клиенты CoAP могут взаимодействовать друг с другом или сервисами в облаке, поддерживающем CoAP. В качестве альтернативы прокси можно использовать для подключения к службе HTTP в облаке. Оконечные точки CoAP могут устанавливать отношения друг с другом даже на уровне датчиков. Наблюдатели поддерживают атрибуты, подобные подписке, для ресурса, который изменяется, подобно MQTT. На графике также показаны серверы происхождения, на которых находятся совместно используемые ресурсы.

Два прокси позволяют CoAP выполнять перевод на HTTP или отправку запросов от имени клиента (рис. 9.7).

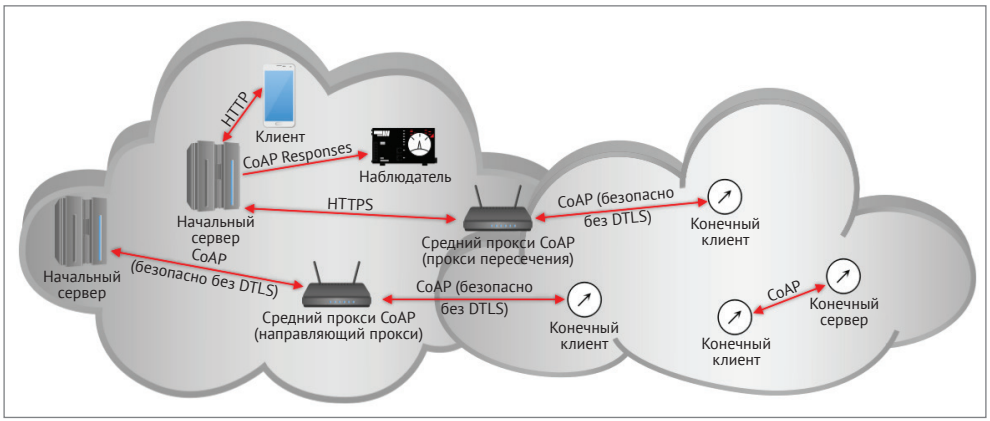


Рис. 9.7 ❖ Архитектура CoAP

CoAP использует порт 5683. Этот порт должен поддерживаться сервером, который предлагает ресурсы, поскольку порт используется для обнаружения ресурсов. Порт 5684 используется, когда DTLS включен.

Форматы сообщений CoAP

Протоколы, основанные на транспорте UDP, подразумевают, что соединение не может быть, по своей сути, надежным. Чтобы компенсировать проблемы надежности, CoAP вводит два типа сообщений, которые отличаются либо требованием подтверждения доставки, либо нет. Дополнительной особенностью этого подхода является то, что сообщения могут быть асинхронными.

В общем, в CoAP всего четыре сообщения:

- **подтверждаемый (CON)** – требуется ACK. Если сообщение CON отправлено, ACK должен быть принят в течение произвольного интервала времени между ACK_TIMEOUT и (ACK_TIMEOUT * ACK_RANDOM_FACTOR). Если ACK не получен, отправитель передает сообщение CON снова и снова с экспоненциально возрастающими интервалами до тех пор, пока не получит ACK или RST. Это, по сути, форма контроля перегрузки CoAP. Максимальное количество попыток установлено с помощью MAX_RETRANSMIT. Это механизм обеспечения отказоустойчивости для компенсации отсутствия таковой в UDP;
- **неподтверждаемый (NON)** – не требуется ACK. По сути, это сообщение «сжечь и забыть» или транслировать широковещательно;
- **подтверждение (ACK)** – подтверждает сообщение CON. Сообщение ACK может повторяться обратно вместе с другими данными;
- **сброс (RST)** – указывает, что получено сообщение CON, но контекст отсутствует. Сообщение RST может повторяться обратно вместе с другими данными.

CoAP – это дизайн RESTful с использованием сообщений запрос/ответ, скомпонованных в сообщениях CoAP. Это позволяет повысить эффективность и сохранение полосы пропускания, как показано на рис. 9.8.

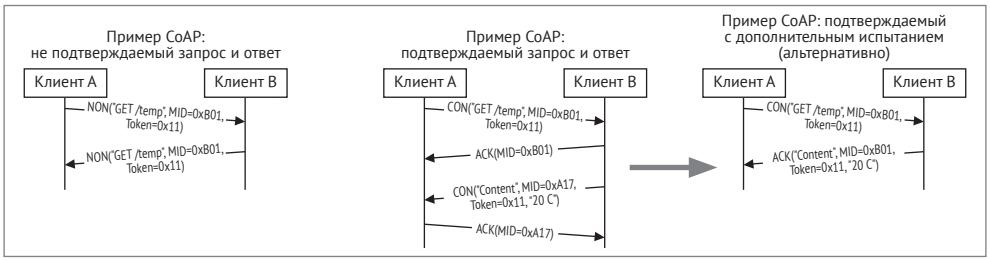
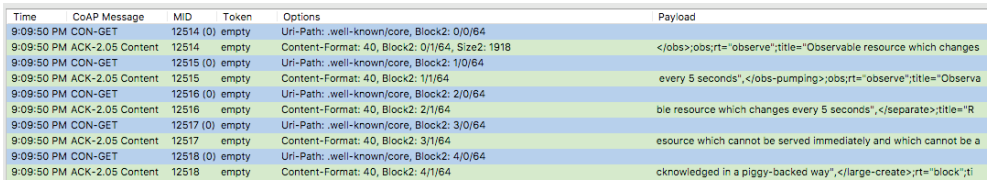


Рис. 9.8 ❖ Сообщения CoAP NON и CON

На графике показаны три примера неподтверждаемых и подтверждаемых транзакций запроса/ответа CoAP. Они определены и описаны следующим образом:

- **неподтверждаемый запрос/ответ (слева)** – сообщение, переданное между клиентами A и B, с использованием типичной конструкции HTTP GET. Через некоторое время B совершает некие манипуляции с данными и возвращает температуру 20 °C;
- **подтверждаемый запрос/ответ (посередине)** – включен идентификатор сообщения, уникальный идентификатор для каждого сообщения. Токен представляет значение, которое должно быть правильным в течение всего периода обмена;
- **подтверждаемый запрос/ответ (справа)** – здесь сообщение может быть подтверждено. Оба клиента A и B будут ждать ACK после каждого обмена сообщениями. Чтобы оптимизировать связь, клиент B может выбрать скомпоновать ACK с возвращаемыми данными, как показано в крайнем правом углу.

Фактический журнал транзакций CoAP можно увидеть в расширении Copper Firefox в Firefox версии 55 (рис. 9.9).



Time	CoAP Message	MID	Token	Options	Payload
9:09:50 PM	CON-GET	12514 (0)	empty	Uri-Path: .well-known/core, Block2: 0/0/64	
9:09:50 PM	ACK-2.05 Content	12514	empty	Content-Format: 40, Block2: 0/1/64, Size2: 1918	</obs>;obs;rt="observe";title="Observable resource which changes
9:09:50 PM	CON-GET	12515 (0)	empty	Uri-Path: .well-known/core, Block2: 1/0/64	
9:09:50 PM	ACK-2.05 Content	12515	empty	Content-Format: 40, Block2: 1/1/64	every 5 seconds",</obs-pumping>;obs;rt="observe";title="Observa
9:09:50 PM	CON-GET	12516 (0)	empty	Uri-Path: .well-known/core, Block2: 2/0/64	
9:09:50 PM	ACK-2.05 Content	12516	empty	Content-Format: 40, Block2: 2/1/64	ble resource which changes every 5 seconds",</separate>;title="R
9:09:50 PM	CON-GET	12517 (0)	empty	Uri-Path: .well-known/core, Block2: 3/0/64	
9:09:50 PM	ACK-2.05 Content	12517	empty	Content-Format: 40, Block2: 3/1/64	esource which cannot be served immediately and which cannot be a
9:09:50 PM	CON-GET	12518 (0)	empty	Uri-Path: .well-known/core, Block2: 4/0/64	
9:09:50 PM	ACK-2.05 Content	12518	empty	Content-Format: 40, Block2: 4/1/64	cknowledged in a piggy-backed way",</large-create>;rt="block";ti

Рис. 9.9 ❖ Журнал Copper CoAP. Здесь мы видим несколько сообщений, инициированных клиентом CON-GET по адресу californium.eclipse.org:5683. Путь URI указывает на coap://californium.eclipse.org: 5683/.wellknown/core. MID увеличивается с каждым сообщением, в то время как токен не используется и не является обязательным

Процесс повторной передачи показан на рис. 9.10.

Хотя для другой архитектуры обмена сообщениями требуется центральный сервер для передачи сообщений между клиентами, CoAP позволяет передавать сообщения между любыми клиентами CoAP, включая датчики и серверы. CoAP включает в себя простую модель кэширования. Кэширование контролируется через коды ответов в заголовке сообщения. Маска номера параметра определяет, является ли это ключом кеша. Параметр `Max_Age` используется для управления временем жизни элемента кэша и обеспечения свежести данных. То есть, `Max_Age` устанавливает максимальное время, в течение которого может быть кэширован ответ, прежде чем он должен быть обновлен. `Max_Age` по умолчанию имеет значение 60 с и может быть расширен до 136,1 года. Прокси играют роль в кэшировании; например, спящий граничный датчик может использовать прокси-сервер для кэширования данных и экономии энергии.

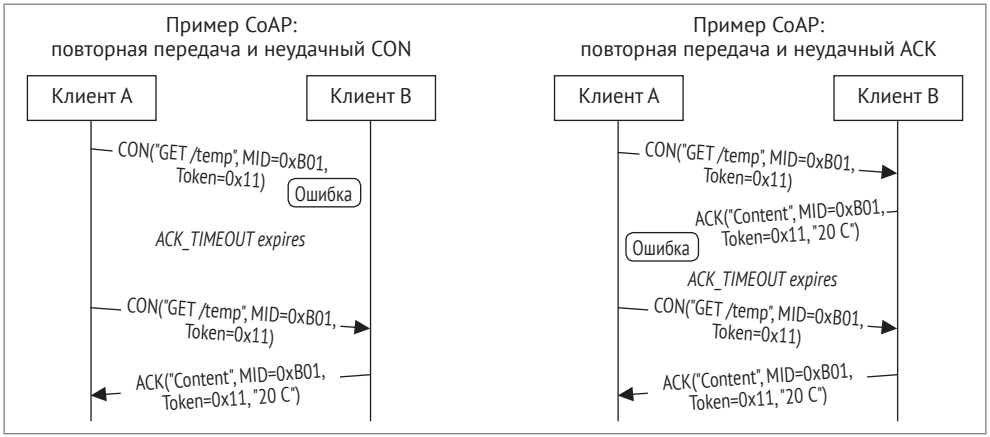


Рис. 9.10 ❖ Механизм повторной передачи CoAP.
Чтобы учесть отсутствие отказоустойчивости в UDP, CoAP использует механизм тайм-аута при общении с подтверждаемыми сообщениями. Если истечет время ожидания, либо отправляя сообщение CON, либо получив ACK, отправитель будет повторно передавать сообщение. Отправитель отвечает за управление таймаутом и повторную передачу до максимального количества повторных передач. Обратите внимание, что повторная передача отказавшего ACK повторно использует тот же Message_ID

Заголовок сообщения CoAP создан для обеспечения максимальной эффективности и сохранения полосы пропускания. Заголовок имеет длину четыре байта с типичным сообщением запроса, содержащим только заголовки длиной от 10 до 20 байт. Это, как правило, в 10 раз меньше, чем заголовок в HTTP. Структура состоит из идентификаторов типа сообщения (T), которые должны быть включены в каждый заголовок вместе с соответствующим уникальным идентификатором сообщения. Поле **Code** используется для сигнализации об ошибках или успешных состояниях по каналам. После заголовка все остальные поля являются необязательными и включают токены переменной длины, опции и полезную нагрузку (рис. 9.11).

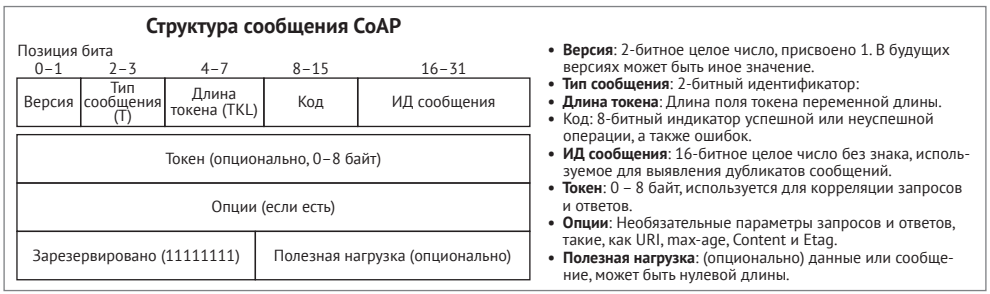


Рис. 9.11 ❖ Структура сообщения CoAP

UDP также может привести к появлению повторяющихся сообщений как для передач CON, так и для NON. Если идентичные `Message_ID` передаются получателю в пределах предписанного `EXCHANGE_LIFETIME`, считается, что есть дубликат. Это может произойти, как показано на предыдущих рисунках, когда отсутствует или сброшен ACK, и клиент повторно передает сообщение с тем же `Message_ID`. В спецификации CoAP указано, что получатель ACK должен получить каждое повторное сообщение, но должен обработать только один запрос или ответ. Это правило может быть упрощено, если сообщение CON передает запрос, который является идемпотентным.

Как уже упоминалось, CoAP допускает роль наблюдателя в системе. Это уникально, так как позволяет CoAP вести себя аналогично MQTT. Процесс наблюдения позволяет клиенту регистрироваться для наблюдения, и сервер будет уведомлять клиента каждый раз, когда контролируемый ресурс изменяет свое состояние. Продолжительность наблюдения может быть определена при регистрации. Кроме того, отношения наблюдения заканчиваются, когда инициирующий клиент отправляет RST или другое сообщение `GET` (рис. 9.12).

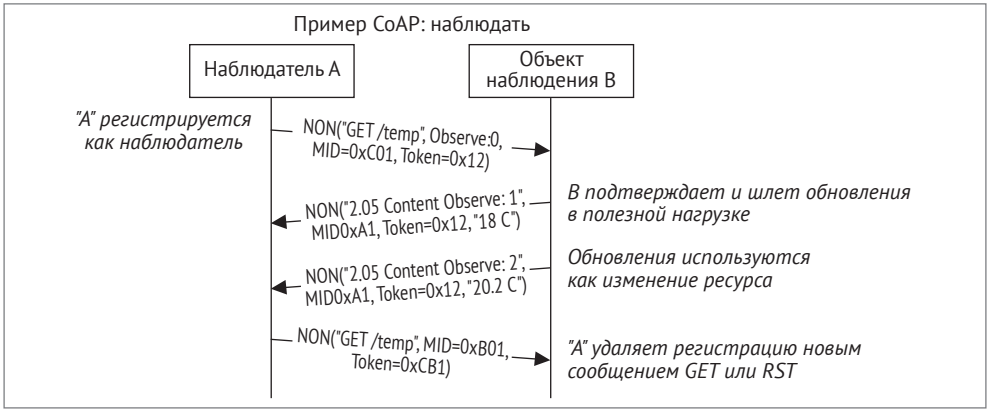


Рис. 9.12 ❖ Процесс регистрации и обновления наблюдателей CoAP

Как упоминалось ранее, в стандарте CoAP нет встроенной проверки подлинности или шифрования, пользователь должен полагаться на DTLS для обеспечения такого уровня безопасности. Если используется DTLS, то пример URI будет следующим:

```
//небезопасно
coap://example.net:1234/~temperature/value.xml

//безопасно
coaps://example.net:1234/~temperature/value.xml
```

CoAP также предлагает механизмы обнаружения ресурсов. Просто отправка запроса `GET` на `/.well-known/core` будет раскрывать список известных ресурсов на

устройстве. Кроме того, строки запроса могут использоваться для применения к запросу определенных фильтров.

Пример использования CoAP

CoAP является легким решением, и его реализация как на клиентской, так и на серверной стороне должна занимать мало ресурсов. Здесь мы используем библиотеку aiocoap на основе Python. Подробнее про aiocoap можно прочитать на: *Amsüss, Christian и Wasilak, Maciej. aiocoap: Библиотека CoAP Python. Энергосберегающие решения, 2013 г.* github.com/chrysn/aiocoap/. Существует множество других бесплатных клиентов и серверов CoAP, несколько написаны на низкоуровневом языке C для чрезвычайно ограниченных сенсорных сред. Здесь мы используем среду Python для краткости.

Реализация клиента:

```
#!/usr/bin/envpython3
importasyncio              #Нужно для асинхронной обработки в Python
fromaiocoapimport *        #с использованием библиотеки aiocoap
```

Ниже приведен основной цикл для клиента. Клиент использует метод PUT для передачи температуры по известному URI:

```
async def main():
 context = await Context.create_client_context()

 awaitasyncio.sleep(2)                              #подождать 2 с после инициализации

 payload = b"20.2 C"
 request = Message(code=PUT, payload=payload)

 request.opt.uri_host = '127.0.0.1'                 #URI локального адреса
 request.opt.uri_path = ("temp", "celcius")         #URI для пути к /temp/Celsius

 response = await context.request(request).response
 print('Result: %s\n%r'%(response.code, response.payload))

if __name__ == "__main__":
 asyncio.get_event_loop().run_until_complete(main())
```

Реализация сервера:

```
#!/usr/bin/envpython3
importasyncio                          #нужно для асинхронной обработки в Python
import aiocoap.resource as resource    #используем библиотеку aiocoap
import aiocoap
```

Следующий код иллюстрирует сервисы для методов PUT и GET:

```
class GetPutResource(resource.Resource):

 def __init__(self):
 super().__init__()
 self.set_content(b"Default Data (padded) "

 def set_content(self, content):                                  #Заполнение
 self.content = content
```

```
 while len(self.content) &lt;= 1024:
 self.content = self.content + b"0123456789\n"

 async def render_get(self, request):                                    #Обработчик GET
 return aiocoap.Message(payload=self.content)

 async def render_put(self, request):                                    #Обработчик PUT
 print('PUT payload: %s' % request.payload)
 self.set_content(request.payload)                                       #замена set_content
                                                                         полученной полезной
                                                                         нагрузкой
 return aiocoap.Message(code=aiocoap.CHANGED, payload=self.content) #set
response code to 2.04
```

Основной цикл:

```
defmain():
 root = resource.Site()         #элемент root, содержащий все ресурсы, найденные на сервере

 root.add_resource(('.well-known', 'core'),                #это типичный ресурс
                                                           .well-known/core
 resource.WKCResource(root.get_resources_as_linkheader))   #список ресурсов для
                                                           .well-known/core
 root.add_resource(('temp', 'celcius'), GetPutResource())) #добавляем ресурс /tmp/celcius

 asyncio.Task(aiocoap.Context.create_server_context(root))

 asyncio.get_event_loop().run_forever()

if __name__ == "__main__":
 main()
```

Другие протоколы

Существует много других протоколов обмена сообщениями, которые используются или предлагаются для развертывания IoT и M2M. Безусловно, наиболее распространенными являются MQTT и CoAP; в следующих разделах рассматриваются несколько альтернатив для специфических случаев использования.

STOMP

STOMP означает простой (или потоковый) протокол промежуточного уровня, ориентированный на текстовые сообщения. Это текстовый протокол, разработанный Codehaus для работы с ориентированным на сообщения промежуточным ПО. Брокер, разработанный на одном языке программирования, может получать сообщения от клиента, написанного на другом языке. Протокол похож на HTTP и работает по TCP. STOMP состоит из заголовка кадра и тела кадра. Текущая спецификация – STOMP 1.2 от 22 октября 2012 г., она доступна по бесплатной лицензии.

Он отличается от многих представленных протоколов, поскольку он не касается вопросов подписки или очередей. Он просто использует семантику, подобную HTTP, такую как SEND со строкой цели. Брокер должен проанализировать со-

общение и сопоставить ему тему или очередь для клиента. Потребитель данных будет ПОДПИСЫВАТЬСЯ на пункты назначения, предоставленные брокером.

У STOMP есть клиенты, написанные на Python (`stomp.py`), TCL (tStomp) и Erlang (`stomp.erl`). Некоторые сервера имеют родную поддержку STOMP, например, RabbitMQ (через плагин), и некоторые серверы были разработаны на специфических языках (Ruby, Perl или OCaml).

AMQP

AMQP означает **Advanced Message Queuing Protocol**. Это стабильный и проверенный протокол MOM, используемый такими крупными источниками данных, как JPMorgan Chase, для обработки более 1 миллиарда сообщений в день и Ocean Observatory Initiative для сбора более 8 терабайт океанографических данных каждый день. Он был первоначально представлен в JPMorgan Chase в 2003 г., которая возглавила создание в 2006 г. рабочей группы из 23 компаний, занимающихся архитектурой и управлением протоколом. В 2011 г. рабочая группа была включена в группу OASIS, где она и находится в настоящее время.

Сегодня протокол хорошо зарекомендовал себя в банковской и кредитной отрасли, но также имеет место в IoT. Кроме того, AMQP стандартизован ISO и IEM как ISO/IEC 19464: 2014. Официальную рабочую группу AMQP можно найти на сайте www.amqp.org.

Протокол AMQP находится вверху стека TCP и использует порт `5672` для связи. Данные сериализуются в AMQP, что означает, что сообщения передаются в единичных кадрах. Кадры передаются по виртуальным каналам, идентифицированных уникальным `channel_id`. Кадры состоят из заголовков, `channel_ids`, информации полезной нагрузки и завершающей части. Канал, однако, может ассоциироваться только с одним хостом. Сообщениям присваивается уникальный глобальный идентификатор.

AMQP – система связи, ориентированная на сообщения и контроль потоков. Это протокол уровня проводов и низкоуровневый интерфейс. Проводной протокол обращается к API сразу над физическим уровнем сети. API-интерфейс на уровне канала позволяет различным службам обмена сообщениями, таким как .NET (NMS) и Java (JMS), взаимодействовать друг с другом. Аналогичным образом AMQP пытается отделить издателей от подписчиков. В отличие от MQTT, он имеет механизмы для балансировки нагрузки и формализации очереди. Широко используемым протоколом, основанным на AMQP, является RabbitMQ. RabbitMQ является брокером сообщений AMQP, написанным на Erlang. Кроме того, доступно несколько клиентов AMQP, таких как клиент RabbitMQ, написанный на Java, C#, Javascript и Erlang, а также Apache Qpid, написанный на Python, C ++, C#, Java и Ruby.

Один или несколько виртуальных хостов с собственными пространствами имен, обменами и очередями сообщений будут находиться на центральном сервере (серверах). Производители и потребители подписываются на услуги обмена. Служба обмена получает сообщения от издателя и направляет данные в соответствующую очередь. Это отношение называется **привязкой**, и привязка может быть либо прямой к одной очереди, либо к нескольким очередям (как

в широковещательной передаче). В качестве альтернативы привязка может ассоциировать один обмен с одной очередью с использованием ключа маршрутизации; это формально называется **прямым обменом**. Другим типом обмена является обмен темы.

Здесь шаблон используется для подстановки ключа маршрутизации (например, `*.temp.#` соответствует `idaho.temp.celsius` и `wisconsin.temp.fahrenheit`) (рис. 9.13).

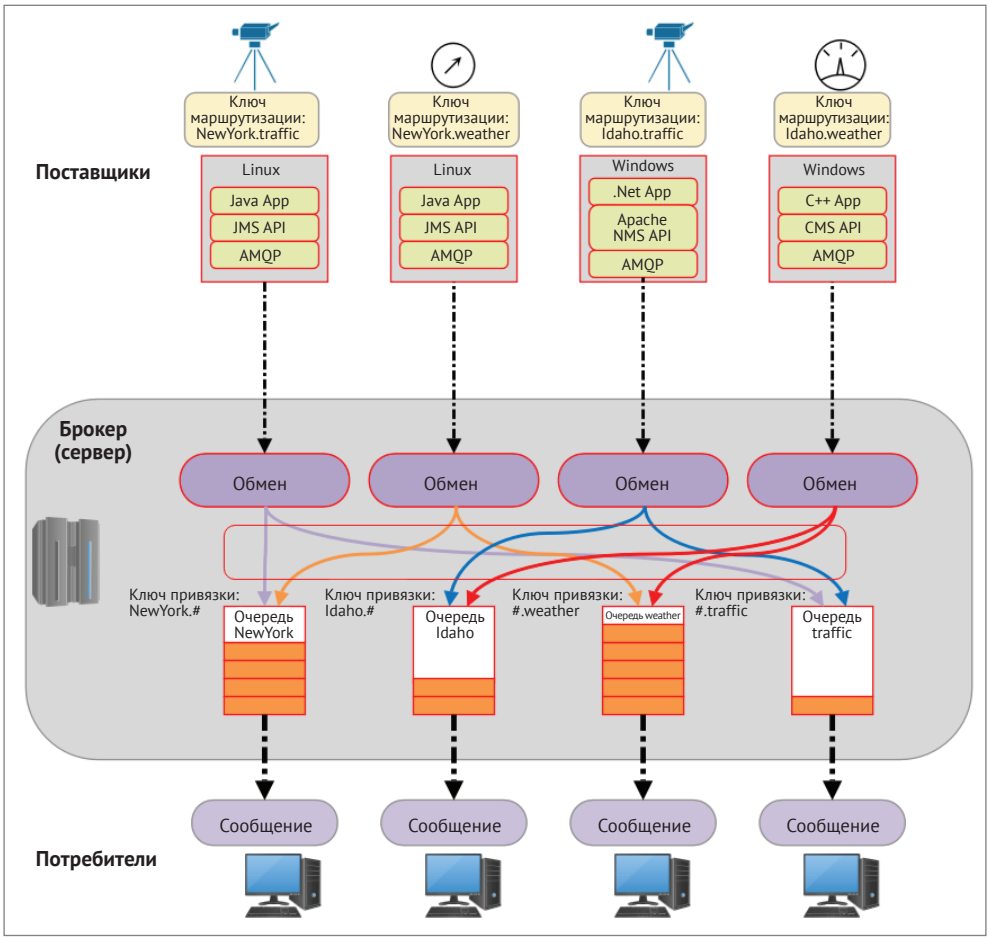


Рис. 9.13 ❖ Архитектурная топология AMQP.
В типичной реализации AMQP есть производители и потребители.
Производители могут использовать разные языки и пространства имен, поскольку AMQP является не зависимым от языка с учетом его API и проводного протокола.
Брокер находится в облачной и обменной сетях для каждого производителя.
Обмены направляют сообщения в соответствующие очереди на основе правил привязки.
Очереди представляют собой буферы сообщений, которые производят сообщения для ожидающих потребителей

Топология сети AMQP основана на принципе «собери и раздай» и предусматривает возможность взаимодействия концентраторов между собой. AMQP состоит из узлов и связей. Узел является именованным источником или стоком сообщений. Кадр сообщения перемещается между узлами по однонаправленным ссылкам. Если сообщение передается через узел, глобальный идентификатор не изменяется. Если узел выполняет любое преобразование, назначается новый идентификатор. Связь имеет возможность фильтровать сообщения. Существует три различных шаблона обмена сообщениями, которые могут использоваться в AMQP:

- **асинхронные направленные сообщения** – сообщение передается без подтверждения приемником;
- **запрос/ответ или pub/sub** – аналогично MQTT с центральным сервером, выступающим в качестве сервиса pub/sub;
- **сохранить и передать** – используется для поддержки концентраторов, где сообщение отправляется на промежуточный концентратор и затем отправляется по направлению к месту назначения.

Здесь показан базовый направленный обмен, написанный на Python, который использует библиотеки RabbitMQ и pika Python. Здесь мы создаем простой прямой обмен под названием **Idaho** и привязываем его к очереди под названием **weather**:

```
#!/usr/bin/env python
#AMQP basic Python example the pika Python library

from pika import BlockingConnection, BasicProperties, ConnectionParameters

#initialize connections
connection = BlockingConnection(ConnectionParameters('localhost'))
channel = connection.channel()
channel.exchange_declare(exchange='Idaho', type='direct')            #объявить прямой обмен
channel.queue_declare(queue='weather')                               #объявить очередь
channel.queue_bind(exchange='Idaho', queue='weather', routing_key='Idaho') #привязка

#создать сообщение
channel.basic_publish(exchange='Idaho', routing_key='Idaho', body='new
important task')

#получить сообщение
method_frame, header_frame, body = ch.basic_get('weather')

#подтвердить
channel.basic_ack(method_frame.delivery_tag)
```

Сводка и сравнение протоколов

Здесь дано резюме и сравнение различных протоколов. Следует отметить, что есть исключения для некоторых из этих категорий. Например, в то время как MQTT не предлагает встроенного средства обеспечения безопасности, его можно задействовать на уровне приложения. Во всех случаях есть исключения, и табл. 9.6 построена по формальным спецификациям.

Таблица 9.6. Сводка и сравнение протоколов

	MQTT	**MQTT-SN**	**CoAP**	**AMQP**	**STOMP**	**HTTP/ RESTful**
Модель	MOM pub/sub	MOM pub/sub	RESTful	MOM	MOM	RESTful
Протокол обнаружения	Нет	Есть (через шлюзы)	Есть	Нет	Нет	Есть
Требования к ресурсам	Низкие	Очень низкие	Очень низкие	Высокие	Средние	Очень высокие
Размер заголовка (байт)	2	2	4	8	8	8
Среднее потребление энергии	Нижайшее	Низкое	Среднее	Высокое	Среднее	Высокое
Аутентификация	Нет (SSL/TLS)	Нет (TLS)	Нет (DTLS)	Есть	Нет	Есть (TLS)
Шифрование	Нет (SSL/TLS)	Нет (SSL/TLS)	Нет (DTLS)	Есть	Нет	Есть (TLS)
Контроль доступа	Нет	Нет	Нет (прокси)	Есть	Нет	Есть
Нагрузка на каналы	Низкая	Очень низкая	Очень низкая	Высокая	Высокая, с проверкой	Высокая
Сложность протокола	Низкая	Низкая	Низкая	Высокая	Низкая	Очень высокая
TCP/UDP	TCP	TCP/UDP	UDP	TCP/UDP	TCP	TCP
Широковещание	Непрямое	Непрямое	Есть	Нет	Нет	Нет
Качество обслуживания	Есть	Есть	С сообщениями CON	Есть	Нет	Нет

Заключение

MQTT, CoAP и HTTP на сегодняшний день являются преобладающими протоколами в отрасли IoT с поддержкой почти всех облачных провайдеров. MQTT и MQTT-SN обеспечивают масштабируемую и эффективную модель передачи данных pub/sub, в то время как CoAP предоставляет все соответствующие возможности модели HTTP RESTful без увеличения накладных расходов. Архитектор должен учитывать накладные расходы, мощность, пропускную способность и ресурсы, необходимые для поддержки данного протокола, и иметь достаточное видение перспективы для обеспечения масштабирования решения. Теперь, когда был определен метод транспорта для направления данных в интернет, мы можем посмотреть, что делать с этими данными. Следующая глава будет посвящена облачной и туманной архитектуре: от основополагающих принципов до более сложных конфигураций.

Глава 10

Топология облачных и туманных вычислений

Без облачных технологий рынок IoT-устройств не существовал бы. По сути, миллиарды оконечных устройств, которые были исторически немыслимы и не связаны, должны были бы самоуправляться, не имея возможности обмениваться данными или собирать данные. Миллиарды небольших встроенных систем не добавляют ценности для клиентов. Значение IoT – в данных, которые он производит, не в одной оконечной точке, но в тысячах и миллионах оконечных точек. Облако обеспечивает возможность иметь простые датчики, камеры, переключатели, маяки и исполнительные механизмы, разговаривающие на общем языке друг с другом. Облако – общий знаменатель валюты данных.

Понятие «облако» относится к инфраструктуре вычислительных служб, которые обычно требуются по запросу. Набор ресурсов (вычислений, сетей, хранилищ и связанных с ними программных сервисов) может динамически масштабироваться в сторону увеличения или уменьшения в зависимости от средней нагрузки и качества обслуживания. Облака, как правило, представляют собой крупные центры обработки данных, которые предоставляют клиентам услуги, ориентированные на внешнего потребителя, и модель оплаты за использование. Эти центры создают иллюзию единого облачного ресурса, в то время как на самом деле может быть использовано много географически распределенных ресурсов. Это дает пользователю чувство независимости от местоположения. Ресурсы являются эластичными (что означает масштабируемость), а сервисы – это эквивалент платы за использование, неизменный доход для провайдера. Сервисы, которые работают в облаке, отличаются от традиционного программного обеспечения своей конструкцией и реализацией. Облачные приложения могут разрабатываться и развертываться быстрее и в меньшей степени зависеть от изменчивости среды. Таким образом, развертывание облаков происходит очень быстро.

Есть сведения, что первое описание облака возникло в Compaq в середине 1990-х гг., где технологические футуристы предсказывали вычислительную

модель, которая переводила вычисления в сеть и на хосты. По сути, это было основой облачных вычислений, но только с появлением некоторых других технологий облачные вычисления стали практичными в отрасли. Телекоммуникационная индустрия традиционно была построена на двухточечной системе связи. Создание виртуальных частных сетей позволило обеспечить безопасный и контролируемый доступ к кластерам и создать частно-публичные облачные гибриды.

В этой главе рассматриваются облачная архитектура и следующие моменты:

- формальное определение облачных топологий и общеупотребительных терминов;
- обзор архитектуры облака OpenStack;
- изучение фундаментальной проблемы чисто облачной архитектуры;
- обзор туманных вычислений;
- справка по архитектуре OpenFog;
- топологии и примеры использования туманных вычислений.

В этой главе мы обсудим несколько случаев практического использования, чтобы вы могли понять влияние семантики больших данных на среду датчиков IoT.

Модель облачных сервисов

Облачные провайдеры обычно поддерживают целый ряд продуктов «**Все как сервис» (XaaS)**. То есть, услуга программного обеспечения с оплатой за использование. Сервис включает в себя **службу сети** (**NaaS**), **программное обеспечение как услугу** (**SaaS**), **платформу как услугу** (**PaaS**) и **инфраструктуру как услугу** (**IaaS**). Каждая модель представляет все больше и больше облачных сервисов от поставщиков. Эти сервисные предложения – добавленная стоимость облачных вычислений. Как минимум, эти услуги должны компенсировать капитальные затраты, с которыми сталкивается клиент для приобретения и обслуживания такого оборудования центра обработки данных, и учесть это как эксплуатационные расходы. Стандартное определение облачных вычислений можно найти в Национальном институте стандартов и технологий: Питер М. Мелл и Тимоти Гранс. SP 800-145. NICE Определение облачных вычислений. Технический отчет. NIST, Gaithersburg, MD, США.

На рис. 10.1 показаны различия в управлении облачными моделями, которые будут описаны в следующих разделах.

NaaS включает такие сервисы, как SDP и SDN. IaaS подталкивает аппаратные системы и хранилище к облаку. PaaS включает в себя инфраструктуру, но также управляет операционной системой и временем выполнения системы или контейнерами в облаке. Наконец, SaaS подталкивает все сервисы, инфраструктуру и сервисы к облачному провайдеру.

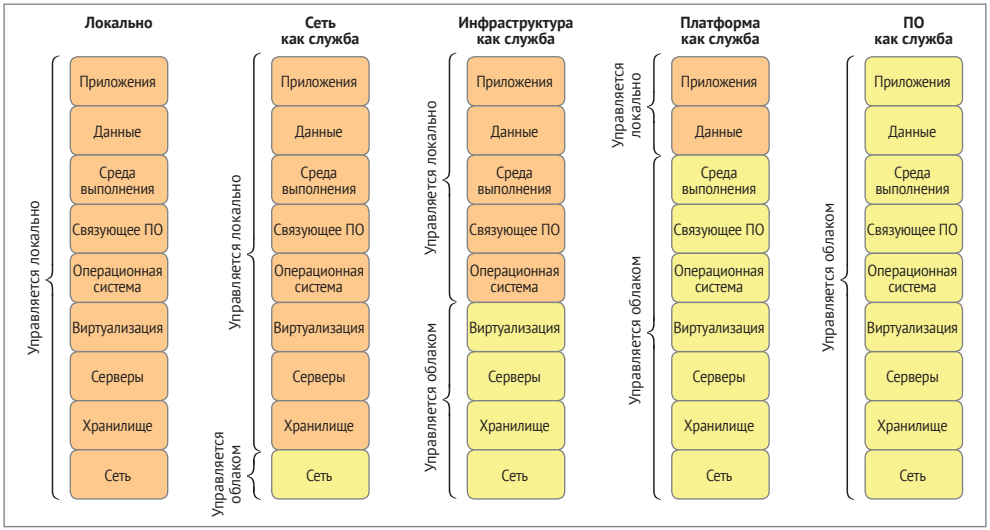


Рис. 10.1 ❖ Модели облачной архитектуры. Собственный объект – это то, где управление всеми службами, инфраструктурой и хранилищем осуществляется владельцем

NaaS

Для NaaS характерны такие сервисы, как **Сетевое взаимодействие, определенное ПО (SDN)** и **Программно-определенные периметры (SDP)**. Эти продукты являются управляемыми облаками и организованными механизмами для обеспечения оверлейных сетей и безопасности предприятий. Вместо того, чтобы создавать глобальную инфраструктуру и выделять капитал для поддержки корпоративных коммуникаций, при создании виртуальной сети может использоваться облачный подход. Это позволяет сети оптимально масштабировать ресурсы в сторону увеличения или уменьшения в зависимости от потребностей, а новые сетевые качества могут быть приобретены и развернуты быстро. Эта тема будет подробно рассмотрена в соответствующей главе SDN.

SaaS

SaaS является основой облачных вычислений. У провайдера обычно есть предлагаемые приложения или услуги, которые предлагаются конечным пользователям с помощью таких клиентов, как мобильные устройства, тонкие клиенты или фреймворки в других облаках. С точки зрения пользователя, виртуальный SaaS фактически работает на клиенте пользователя. Эта абстракция программного обеспечения позволила отрасли добиться значительного роста в облачном сервисе. Сервисы SaaS работают для таких устройств, как Google Apps, Salesforce и Microsoft Office 365.

PaaS

PaaS использует базовое оборудование и программные средства нижнего уровня, предоставляемые облаком. В таком случае конечный пользователь просто использует аппаратное обеспечение центра обработки данных, операционную систему, промежуточное ПО и различные базы данных поставщика для размещения своего частного приложения или сервисов. Промежуточное ПО может состоять из систем баз данных. При построении многих отраслей промышленности было использовано оборудование облачных поставщиков, например для Swedbank, Trek Bicycles и Toshiba. Примерами публичных поставщиков PaaS являются IBM Bluemix, Google App Engine и Microsoft Azure. Разница между PaaS и IaaS заключается в том, что вы получаете преимущества масштабируемости и OPEX с облачной инфраструктурой, но у вас также есть проверенное промежуточное ПО и операционные системы от провайдера. Это такие системы, как Docker, где программное обеспечение развертывается в контейнерах. Если ваше приложение развертывается в пределах ограничений предоставляемой поставщиком инфраструктуры, вы можете ожидать более быстрый выход на рынок, поскольку большинство компонентов, ОС и промежуточного программного обеспечения гарантированно будут доступны.

IaaS

IaaS была изначальной концепцией облачных сервисов. В этой модели поставщик создает масштабируемые аппаратные службы в облаке и предоставляет модификацию программных фреймворков для создания клиентских виртуальных машин. Это обеспечивает максимальную гибкость при развертывании, но требует бóльших усилий со стороны клиента.

Публичное, частное и гибридное облако

В облачной среде существуют три различные модели топологии облаков, которые обычно используются: частное облако, облако общего пользования и гибридное облако. Независимо от модели, фреймворки облаков должны обеспечивать динамическую масштабируемость, быстроту разработки и развертывания, а также появление в локальном месте независимо от его близости (рис. 10.2).

Частные облака также подразумевают управляемые компоненты по запросу. Современные корпоративные системы, как правило, используют гибридную архитектуру для обеспечения безопасности критически важных приложений и данных на местности и используют публичное облако для подключения, простоты и быстроты развертывания.

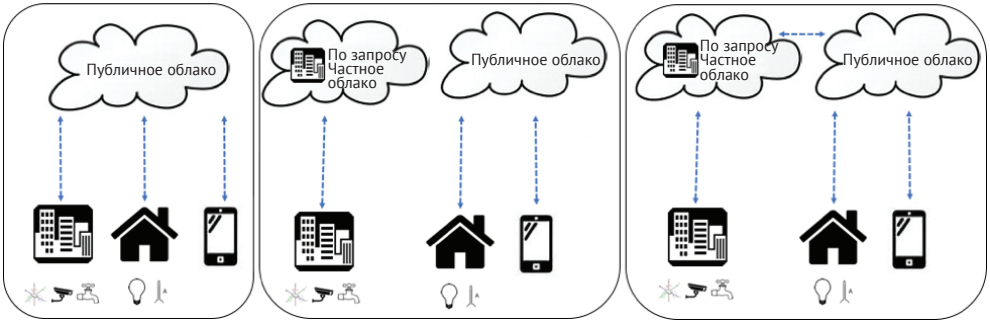


Рис. 10.2 ❖ Слева: публичное облако.
Посередине: частное против публичного. Справа: гибридное облако

Частное облако

В частном облаке инфраструктура предоставлена одной организации или корпорации. Нет концепции совместного использования ресурсов или объединения за пределами собственной инфраструктуры владельца. В помещениях совместное использование и распоряжение ресурсами являются общими. Частное облако существует по ряду причин, включая безопасность и проверенность качества. То есть, для гарантии, что информация обрабатывается исключительно системами, управляемыми клиентом. Однако, чтобы считаться облаком, должны существовать некоторые аспекты облачных сервисов, такие как виртуализация и балансировка нагрузки. Частное облако может быть локальным или может быть специализированным в оборудовании, предоставляемым третьей стороной исключительно для его использования.

Публичное облако

Публичное облако – противоположная ситуация. Здесь инфраструктура предоставляется по требованию для множества клиентов и приложений. Инфраструктура представляет собой набор ресурсов, которые любой человек может использовать в любое время в рамках своих соглашений об уровне обслуживания. Преимущество здесь в том, что явная шкала облачных центров обработки данных позволяет обеспечить беспрецедентную масштабируемость для многих клиентов, которые ограничены только тем, какую часть услуг они хотят приобрести.

Гибридное облако

Гибридная архитектурная модель представляет собой сочетание частных и общественных облаков. Такими комбинациями могут быть множественные публичные облака, используемые одновременно или комбинация общественной и частной облачной инфраструктуры. Организации предпочитают гибридную модель, если есть данные, которые нуждаются в уникальном подходе, а интер-

фейс может использовать облако. Другим вариантом использования является поддержание соглашения с облачными областями для компенсации условий, когда масштабируемость лучше, чем у частной корпорации в целом. В этом случае публичное облако будет использоваться как балансировщик нагрузки до тех пор, пока набирание данных и их использование не вернутся в ограниченное пространство частного облака. Этот вариант использования называется облачным взрывом и относится к использованию облаков в качестве условных ресурсов.

Облачная архитектура OpenStack

OpenStack – это сервер Apache 2.0 с открытым исходным кодом, используемый для создания облачных платформ. Это IaaS разрабатывается сообществом разработчиков с 2010 г. OpenStack Foundation управляет программным обеспечением и поддерживает более 500 компаний, включая Intel, IBM, Red Hat и Ericsson. Мы будем использовать OpenStack в качестве эталонной архитектуры для других поставщиков облачных вычислений, поскольку большая часть компонентов и терминология также используются в коммерческих облаках.

OpenStack начинался как совместный проект NASA и Rackspace в 2010 г. Архитектура имеет все основные компоненты других облачных систем, включая вычисление и балансировку нагрузки; компоненты хранения, включая резервное копирование и восстановление; сетевые компоненты, информационные панели, системы безопасности и идентификации, пакеты данных и аналитики, инструменты развертывания, мониторы, счетчики и приложения. Это те компоненты, которые будет использовать архитектор при выборе облачного сервиса.

С точки зрения архитектуры, OpenStack представляет собой смешанные слои компонентов. Основная форма облака OpenStack показана на рис. 10.3. Каждый сервис имеет определенную функцию и уникальное имя (например, Nova). Система работает, в целом, предоставляя масштабируемые функциональные возможности облачного класса масштаба корпорации.

Все коммуникации в компонентах OpenStack выполняются через **протокол расширенной очереди сообщений (AMQP)**, в частности, RabbitMQ или Qpid. Сообщения могут быть либо неблокирующими, либо блокирующими в зависимости от того, как было отправлено сообщение. Сообщение будет отправлено как объект JSON в RabbitMQ, и получатели получат свои сообщения в одном сервисе. Это метод связи (**Remote Procedure Call – RPC**) между основными подсистемами. Преимущество облачной среды заключается в том, что проблемы клиента и сервера полностью независимы друг от друга, и это позволяет серверам динамически масштабироваться в сторону увеличения или уменьшения. Сообщения не передаются, а направляются, что снижает трафик до минимума. Вы, возможно, помните, что AMQP – это стандартный протокол обмена сообщениями, используемый в пространстве IoT.

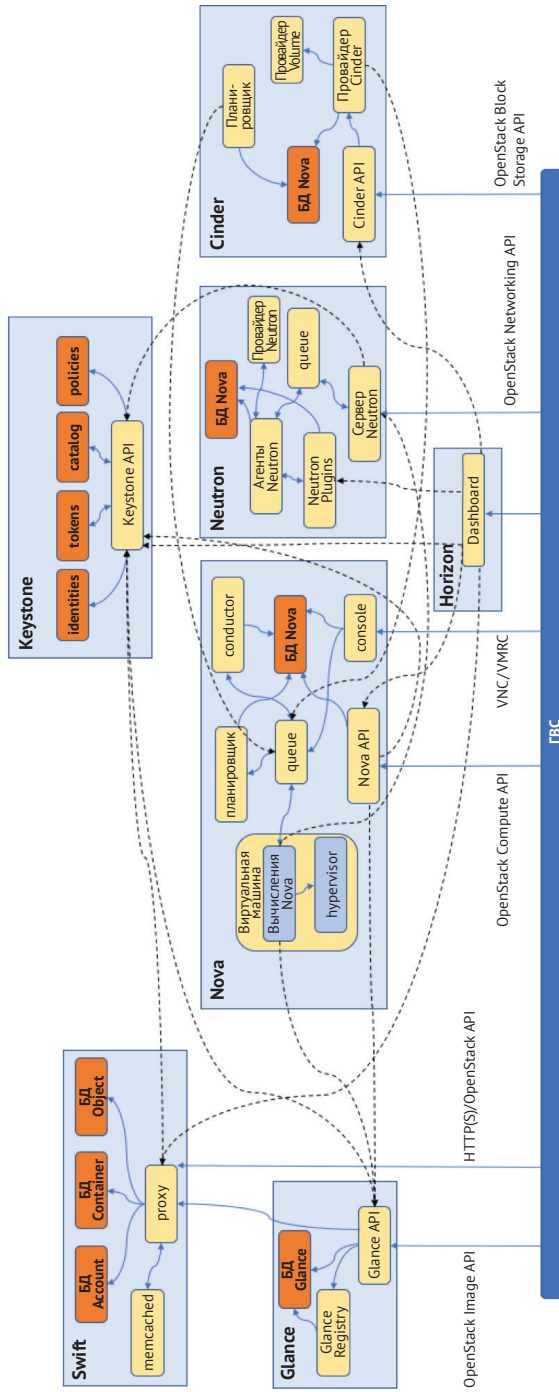


Рис. 10.3 ❖ Высокоуровневая архитектурная диаграмма OpenStack

Keystone – управление идентификацией и обслуживанием

Keystone – это служба управления идентификаторами облака OpenStack. Менеджер идентификации устанавливает учетные данные пользователя и авторизацию для входа. Это, по сути, отправная точка или точка входа в облако. Этот ресурс будет поддерживать центральный каталог пользователей и их прав доступа. Это самый высокий уровень безопасности, обеспечивающий независимость и безопасность пользовательских сред. Keystone может взаимодействовать с такими сервисами, как LDAP, на корпоративном уровне. Keystone также поддерживает базу данных токенов и предоставляет временные токены пользователям аналогично тому, как **Amazon Web Services (AWS)** устанавливает учетные данные. Реестр служб используется для запроса продуктов или услуг, доступных пользователю программно.

Glance – сервис изображений

Glance – это сердцевина управления виртуальными машинами для OpenStack. Большинство облачных сервисов обеспечит некоторую степень виртуализации и будет иметь аналоговый ресурс, подобный Glance. API службы изображений – это служба RESTful, что позволяет клиенту разрабатывать шаблоны VM, обнаруживать доступные виртуальные машины, клонировать изображения на другие серверы, регистрировать виртуальные машины и даже беспрепятственно перемещать работающие виртуальные машины на разные физические серверы без перерыва в работе. Glance вызывает Swift (хранилище объектов) для извлечения или хранения различных изображений. Glance поддерживает разные стили виртуальных образов:

- `raw` – неструктурированные изображения;
- `vhd` – VMWare, Xen, OracleVirtualBox;
- `vmdk` – общий формат диска;
- `vdi` – изображение эмулятора QEMU;
- `iso` – изображение на оптическом диске (CD-ROM);
- `aki/ari/ami` – изображение Amazon.

Виртуальная машина состоит из всего содержимого образа жесткого диска, включая гостевые операционные системы, среды выполнения, приложения и службы.

Вычисления Nova

Это основа службы управления вычислительными ресурсами OpenStack. Ее цель – определить и учесть вычислительные ресурсы на основе спроса. Она также несет ответственность за управление системным гипервизором и виртуальными машинами. Nova может работать с несколькими виртуальными машинами, например с VMware или Xen, или может управлять контейнерами. Масштабирование по требованию является неотъемлемой частью любого облачного предложения.

Nova основана на API веб-службы RESTful для упрощения управления.

Чтобы получить список серверов, можно выполнить следующее в Nova через запрос API `GET`:

```
{url_вашей_вычислительной_службы}/servers
```

Чтобы создать банк серверов (в пределах: минимум 10 и максимум 20), мы используем метод `POST`:

```
{
    "server": {
        "name": "IoT-Server-Array",
        "imageRef": "8a9a114e-71e1-aa7e-4181-92cc41c72721",
        "flavorRef": "1",
        "metadata": {
            "My Server Name": "IoT"
        },
        "return_reservation_id": "True",
        "min_count": "10",
        "max_count": "20"
    }
}
```

Nova в ответ выдаст `reservation_id`:

```
{
    "reservation_id": "84.urcyplh"
}
```

Таким образом, модель программирования достаточно проста для управления инфраструктурой.

База данных Nova необходима для поддержания текущего состояния всех объектов в кластере. Например, несколько состояний различных серверов в кластере могут быть следующими:

- `ACTIVE` – сервер активно работает;
- `BUILD` – сервер в состоянии сборки и пока не закончен;
- `DELETED` – сервер был удален;
- `MIGRATING` – сервер переносится на другой хост.

Nova полагается на планировщик, чтобы определить, какую задачу выполнить и где ее выполнить. Планировщик может случайно ассоциировать сродство или использовать фильтры, чтобы выбрать набор хостов, которые наилучшим образом соответствуют некоторым наборам параметров. Конечным продуктом фильтра будет упорядоченный список хост-серверов для использования от лучшего к худшему (несовместимые хосты будут удалены из списка).

Ниже приведен фильтр по умолчанию, используемый для выбора родственного сервера:

```
scheduler_available_filters = nova.scheduler.filters.all_filters
```

Пользовательский фильтр может быть создан (например, Python или JSON-фильтр с именем `IoTFilter.IoTFilter`) и прикреплен к планировщику следующим образом:

```
scheduler_available_filters = IoTFilter.IoTFilter
```

Чтобы установить фильтр для поиска серверов, имеющих 16 VCPU программным путем через API, мы создаем файл JSON следующим образом:

```
{
    "server": {
        "name": "IoT_16",
        "imageRef": "8a9a114e-71e1-aa7e-4181-92cc41c72721",
        "flavorRef": "1"
    },
    "os:scheduler_hints": {
        "query": "[&gt;=,$vcpus_used,16]"
    }
}
```

Как вариант, OpenStack также позволяет управлять облаком через интерфейс командной строки:

```
$ openstack server create --image 8a9a114e-71e1-aa7e-4181-92cc41c72721 \
  --flavor 1 --hint query='["&gt;=","$vcpus_used",16]' IoT_16
```

OpenStack имеет богатый набор фильтров, позволяющих настраивать распределение серверов и сервисов. Это позволяет очень четко контролировать обеспечение и масштабирование сервера. Это классический и очень важный аспект облачного дизайна. Такие фильтры включают, но не ограничиваются такими характеристиками, как:

- размер оперативной памяти;
- емкость и тип диска;
- уровни IOPS;
- использование процессора;
- групповая родственность;
- родственность с CIDR.

Swift – хранение объектов

Swift предоставляет резервную систему хранения для центра обработки данных OpenStack. Swift позволяет масштабировать кластеры путем добавления новых серверов. Хранилище объектов будет содержать такие вещи, как учетные записи и контейнеры. Виртуальная машина пользователя может храниться или кэшироваться в Swift. Вычислительный узел Nova может вызывать непосредственно Swift и загружать изображение при первом запуске.

Neutron – сетевые сервисы

Neutron – это управление сетью OpenStack и служба VLAN. Вся сеть является настраиваемой и предоставляет такие услуги, как:

- доменные службы имен;
- DHCP – протокол динамической конфигурации хостов;

- функции шлюза;
- управление VLAN;
- соединения на втором уровне модели OSI;
- SDN;
- протоколы с покрытием и туннелированием;
- VPN;
- NAT (SNAT и DNAT);
- системы обнаружения вторжений;
- балансировка нагрузки;
- брандмауэры.

Cinder – блочное хранилище

Cinder обеспечивает OpenStack постоянными службами хранения блоков, необходимыми для облака. Он выступает в роли *хранилища как службы* для использования с базами данных, динамическими файловыми системами и в других случаях, где важна защита от утечек данных. Это важно и для потоковых сценариев IoT. Как и другие компоненты OpenStack, система хранения сама по себе динамична и масштабируется по мере необходимости. Архитектура построена на принципах высокой доступности и открытых стандартах.

Функциональность, предоставляемая Cinder, включает:

- создание, удаление и привязку устройств хранения к экземплярам Nova;
- совместимость с несколькими хранилищами (HP 3PAR, EMC, IBM, Ceph, CloudByte, Scality);
- поддержку нескольких интерфейсов (Fibre Channel, NFS, Shared SAS, IBM GPFS, iSCSI);
- резервное копирование и извлечение образов дисков;
- сохранение изображений в определенные моменты времени;
- альтернативное хранилище для изображений VM.

Horizon

Последний элемент, рассматриваемый здесь – Horizon. Horizon – это панель инструментов OpenStack. Это упрощенный вид в OpenStack для клиента. Он обеспечивает веб-представление различных компонентов, которые включают OpenStack (Nova, Cinder, Neutron и другие). Horizon представляет собой изображение пользовательского интерфейса облачной системы в качестве альтернативного средства поверх API. Horizon расширяем, поэтому третья сторона может добавлять свои виджеты или инструменты в панель инструментов. Можно добавить новый компонент биллинга, и затем для клиентов может быть создан соответствующий элемент панели Horizon.

Большинство систем IoT, которые используют облачные вычисления, будут иметь некоторую форму панели мониторинга с аналогичными функциями.

Heat – оркестрация (опция)

Heat может запускать несколько составных облачных приложений и управлять облачной инфраструктурой на основе шаблонов в экземпляре OpenStack. Heat интегрируется с телеметрией для автоматической настройки системы в соответствии с нагрузкой. Шаблоны в Heat пытаются соответствовать форматам AWS CloudFormation, а отношения между ресурсами могут быть указаны аналогичным образом (например, данный том подключен к данному серверу).

Шаблон Heat может быть похож на следующее:

```
heat_template_version: 2015-04-30

description: example template

resources:
  my_instance:
    type: OS::Nova::Server
    properties:
      key_name: { get_param: key_name }
      image: { get_param: image }
      flavor: { get_param: flavor }
      admin_pass: { get_param: admin_pass }
      user_data:
        str_replace:
          template: |
            #!/bin/bash
            echo hello_world
```

Ceilometer – телеметрия (опция)

OpenStack предоставляет дополнительный сервис под названием Ceilometer, который может использоваться для сбора данных телеметрии и учета ресурсов, используемых каждой службой. Измерение используется для сбора информации об использовании и преобразования его в счета клиента. Ceilometer также предоставляет инструменты оценки и выставления счетов. Значение выставленной стоимости конвертируется в эквивалентную валюту, а биллинг используется для начала процесса оплаты.

Ceilometer контролирует и измеряет различные события, такие как запуск службы, добавление тома и остановка экземпляра. Метрики собираются по использованию ЦПУ, количеству ядер, использованию памяти и перемещению данных. Все это собирается и хранится в базе данных MongoDB.

Ограничения облачных архитектур для IoT

Поставщик облачных сервисов находится за пределами граничного устройства IoT и руководит глобальной сетью. Одной из особенностей архитектуры IoT является то, что устройства PAN и WPAN могут не соответствовать протоколу IP. Протоколы, такие как **Bluetooth Low Energy (BLE)** и Zigbee, не основаны на IP, тогда как все в глобальных сетях, включая облака, основано на IP. Таким

образом, роль пограничного шлюза заключается в выполнении перевода из одного протокола в другой (рис. 10.4).

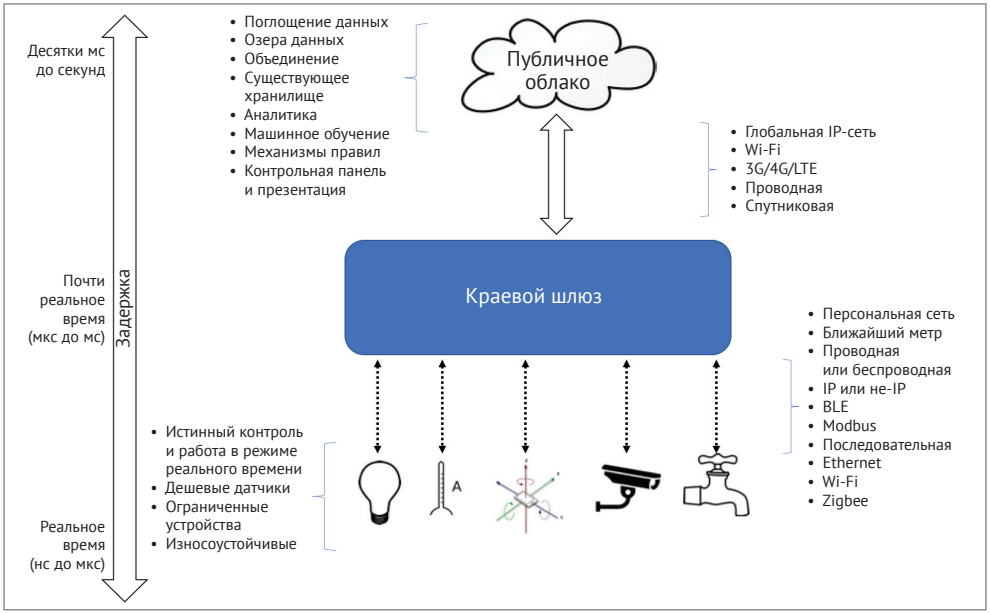


Рис. 10.4 ❖ Эффекты задержек в облаке.
Реакция в реальном времени имеет решающее значение во многих приложениях IoT и вынуждает передвигать обработку ближе к оконечному устройству

Эффект задержки

Другим эффектом является время ожидания и время отклика для событий. По мере приближения к датчику вы входите в область, где действуют жесткие требования выполнения в реальном времени. Эти системы, как правило, представляют собой глубоко встроенные системы или микроконтроллеры с задержкой, предназначенные для реальных событий. Например, видеокамера чувствительна к частоте кадров (как правило, 30 или 60 кадров в секунду) и должна выполнять ряд последовательных задач в конвейере потока данных (избавление от мозаики, обозначение, баланс белого и гамма-регулирование, отображение гаммы, масштабирование и сжатие). Объем данных, проходящих через конвейер видеоизображения (видео 1080p с использованием 8 бит на канал со скоростью 60 кадров в секунду) составляет примерно 1,5 ГБ/с. Каждый кадр должен проходить через этот конвейер в режиме реального времени, поэтому большинство процессоров сигналов видеоизображения используют для этих преобразований контроллеры.

Если мы переместимся вверх по стеку, шлюз будет иметь лучшее время отклика и обычно реагирует за миллисекунды с одной цифрой. Коэффициент

стробирования во времени отклика – это латентность WPAN и нагрузка на шлюз. Как упоминалось ранее в главе о WPAN, большинство WPAN, таких как BLE, являются переменными и зависят от количества устройств BLE под шлюзом, интервалов сканирования, интервалов рекламы и т. д. Интервалы соединения BLE могут достигать 7,5 мс, но могут варьироваться в зависимости от того, как клиент настраивает интервалы рекламы, чтобы свести к минимуму потребление энергии. Сигналы Wi-Fi обычно имеют задержку 1,5 мс. Задержка такого уровня требует физического интерфейса с PAN. Нельзя ожидать, что при передаче необработанных пакетов BLE в облако, будет быстродействие почти в реальном времени.

Компонент обработки в облаке добавляет еще одну степень задержки в WAN. Маршрут между шлюзом и провайдером облачных вычислений может пролегать несколькими путями на основе расположения центров обработки данных и шлюза. Облачные провайдеры обычно предоставляют набор региональных центров обработки данных для нормализации трафика. Чтобы понять истинное влияние провайдера облачных вычислений на латентность, нужно отследить задержку пинга в течение недель или месяцев и по регионам (см. табл. 10.1).

Таблица 10.1. Задержка пинга по регионам

Регион	Задержка
US East (Virginia)	80 мс
US East (Ohio)	87 мс
US West (California)	48 мс
US West (Oregon)	39 мс
Canada (Central)	75 мс
Europe (Ireland)	147 мс
Europe (London)	140 мс
Europe (Frankfurt)	152 мс
Asia Pacific (Mumbai)	307 мс
Asia Pacific (Seoul)	192 мс
Asia Pacific (Singapore)	306 мс
Asia Pacific (Sydney)	205 мс
Asia Pacific (Tokyo)	149 мс
South America (São Paulo)	334 мс
China (Beijing)	436 мс
GovCloud (US)	39 мс

Amazon AWS CloudPing от US West Client любезно предоставлен CloudPing.info (для получения дополнительной информации посетите www.cloudping.info).

Исчерпывающий анализ времени задержки для облаков и времени отклика поддерживается CL Audit (для получения дополнительной информации посетите сайт claudit.feld.cvut.cz/index.php#). Существуют и другие инструменты для

анализа латентности, такие как Fathom и SmokePing (дополнительную информацию можно найти на странице oss.oetiker.ch/smokeping/). Эти сайты исследуют, мониторят и сохраняют задержку TCP, HTTP и SQL при работе с базой данных в AWS и Microsoft Azure на ежедневной основе во многих регионах мира. Это обеспечивает наилучшую видимость общего воздействия задержки, которое можно ожидать от облачного решения. Например, приведенная здесь цифра иллюстрирует **время прохождения в оба конца (RTT)** в течение одного дня между тестовым клиентом в США, который связывается с серверами Amazon AWS и Microsoft Azure в западных штатах США. Это также полезно, чтобы отметить изменчивость RTT. Хотя всплеск 5 мс может быть допустимым во многих приложениях, это может привести к сбою в жесткой системе управления в реальном времени или автоматизации производства (рис. 10.5).

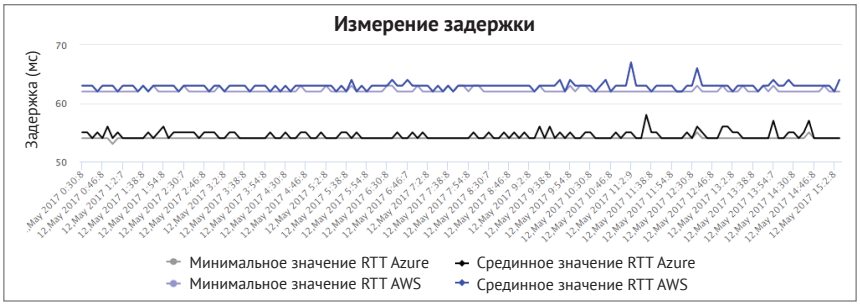


Рис. 10.5 ❖ Время прохождения и задержка для облачных провайдеров. График иллюстрирует латентность в миллисекундах для двух облачных провайдеров в течение нескольких часов

Как правило, задержки в облаке будут составлять десятки, если не сотни миллисекунд, без учета каких-либо накладных расходов на обработку поступающих данных. Теперь это должно быть принято во внимание для различных уровней ответа при построении облачной архитектуры для IoT. Архитектуры близких устройств допускают ответы до 10 мс, а также они повторяемы и детерминированы. Облачные решения могут иметь изменчивое время отклика, а также время задержки на порядок больше, чем при использовании граничных устройств. Архитектор должен учитывать, где развернуть часть решения на основе рассмотрения этих двух эффектов. Поставщиков облачных вычислений также следует выбирать на основе моделей развертывания центров обработки данных. Если решение IoT развертывается во всем мире или, возможно, будет расширяться для охвата нескольких регионов, облачный сервис должен иметь центры обработки данных, расположенные в географически близких областях, чтобы помочь в уменьшении времени отклика. На диаграмме видна большая разница в задержке для одного клиента, связывающегося с центрами обработки данных по всему миру. Это не оптимальная архитектура.

Туманные вычисления

Туманные вычисления – это эволюционное расширение облачных вычислений на краю. В этом разделе описывается разница между вычислениями Fog и Edge и представлены различные топологии и архитектурные ссылки для Fog Computing.

Философия Hadoop для туманных вычислений

Туманные вычисления развили успех Hadoop и MapReduce, и для того, чтобы лучше понять важность Fog Computing, стоит рассмотреть, как работает Hadoop. MapReduce – это метод сопоставления, а Hadoop – это платформа с открытым исходным кодом, основанная на алгоритме MapReduce.

MapReduce включает три шага: отображение, перетасовка и уменьшение. На фазе **отображения** вычислительные функции работают с локальными данным. Шаг перетасовки перераспределяет данные по мере необходимости. Это критический шаг, когда система пытается совместить все зависимые данные в одном узле. Последним шагом является фаза уменьшения, при которой обработка происходит параллельно на всех узлах.

Общий вывод здесь заключается в том, что MapReduce пытается приблизить обработку туда, где находятся данные, и не перемещать данные туда, где находятся процессоры. Эта схема эффективно устраняет перегрузку коммуникаций и естественное узкое место в системах с чрезвычайно большими структурированными или неструктурированными наборами данных. Эта парадигма также применима и к IoT. В пространстве IoT данные (возможно, очень большой объем данных) создаются в реальном времени в виде потока данных. Это данные большого объема в случае IoT. Это не статические данные, такие как база данных или кластерное хранилище Google, а бесконечный поток данных из всех уголков мира. Туманные конструкции – естественный способ решить эту новую проблему с большими данными.

Сравнение туманных, граничных и облачных вычислений

Мы уже определили граничные вычисления как перемещающуюся обработку, близкую к тому месту, где генерируются данные. В случае IoT граничным устройством может быть сам датчик с небольшим микроконтроллером или встроенной системой, способной к WAN-связи. В других случаях граница будет шлюзом в архитектурах с особенно ограниченными оконечными точками, заставляющими шлюз зависать. Граничная обработка также обычно упоминается в контексте «машина-машина», где существует плотная корреляция между краем (клиентом) и сервером, расположенным в другом месте. Граничные вычисления существуют, как указано, для устранения проблем с задержкой и ненужной загрузкой полосы пропускания, а также для добавления таких сервисов, как денатурация и безопасность, близким к источнику данных. У граничного устройства может быть связь с облачным сервисом

ценой задержки и несущей; оно не принимает активного участия в облачной инфраструктуре.

Облачные вычисления немного отличаются от парадигмы граничных вычислений. В облачных вычислениях в первую очередь разделяется API инфраструктуры и стандарты связи с другими туманными узлами и/или покрывающей облачной службой. Туманные узлы являются расширениями облака, тогда как граничные устройства могут использовать или не использовать облако. Другим ключевым принципом туманных вычислений является то, что туман может существовать в иерархических слоях. Туманные вычисления также могут балансировать нагрузку и управлять данными с востока на запад и с севера на юг, чтобы помочь в балансировке ресурсов. Из определения облака и его сервисов, которые были описаны в предыдущем разделе, можно подумать об этих туманных узлах как просто о еще одних (хотя и менее мощных) инфраструктурах в гибридном облаке.

Архитектура OpenFog RA

Архитектура туманного фреймворка, такого как облачный фреймворк, необходима для понимания межсетевого взаимодействия и контрактов по данным между различными уровнями. Здесь мы изучаем OpenFog Consortium Reference Architecture: www.openfogconsortium.org/wp-content/uploads/OpenFog_Reference_Architecture_2_09_17-FINAL.pdf. Консорциум OpenFog является некоммерческой отраслевой группой, занятой определением стандартов совместимости с туманными вычислениями. Хотя они не являются органом по стандартизации, но влияют на направление деятельности других организаций посредством взаимодействия и влияния в отрасли. Справочная архитектура OpenFog – это модель, помогающая архитекторам и бизнес-лидерам в создании оборудования, программного обеспечения и приобретении инфраструктуры для туманных вычислений. OpenFog понимает преимущества облачных решений и желание расширить этот уровень вычислений, хранения, сетей и масштабируемости до граничного уровня, не жертвуя задержками и пропускной способностью.

OpenFog Reference Architecture представляет собой многослойный подход от граничных датчиков и исполнительных механизмов внизу, к приложениям наверху. Архитектура имеет некоторое сходство с типичной облачной архитектурой, такой как OpenStack, но расширяет ее, поскольку она более похожа на PaaS, чем на IaaS. В этом случае OpenFog предоставляет полный стек и, как правило, аппаратно независим или, по крайней мере, абстрагирует платформу от остальной части системы (рис. 10.6).

Службы приложений

Роль сервисного уровня заключается в предоставлении «оконного стекла» и специальных услуг, необходимых для выполнения задачи. Это включает в себя предоставление соединителей для других служб, содержание пакетов

для аналитики данных, предоставление пользовательского интерфейса при необходимости и предоставление основных услуг.

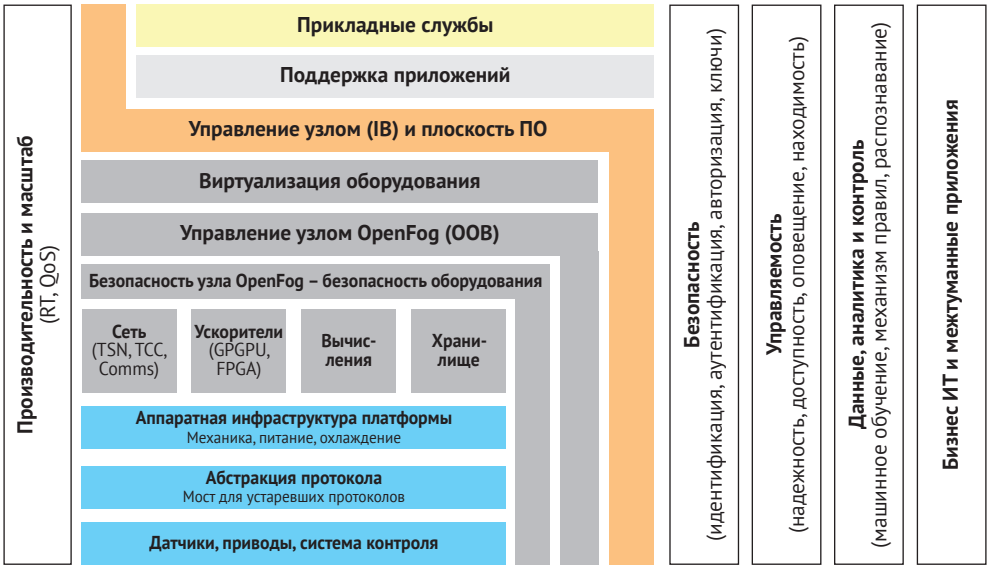


Рис. 10.6 ❖ Архитектура OpenFog RA

Соединители на прикладном уровне соединяют службы с уровнем поддержки. Уровень абстракции протокола обеспечивает путь для общения соединителя непосредственно с датчиком. Каждая услуга должна рассматриваться как микросервис в контейнере. Консорциум OpenFog выступает за метод развертывания контейнеров в качестве основного метода развертывания программного обеспечения на границе. Это имеет смысл, когда мы рассматриваем граничные устройства как расширения облака. Пример развертывания контейнера может выглядеть следующим образом. Каждый цилиндр представляет собой отдельный контейнер, который можно развернуть и управлять им отдельно. Затем каждый сервис предоставляет API для взаимодействия контейнеров и слоев (рис. 10.7).

Поддержка приложения

Это компонент инфраструктуры, который поможет собрать окончательное решение для клиентов. Этот уровень может иметь зависимости с точки зрения того, как он был развернут (например, как контейнер). Поддержка осуществляется во многих формах, в том числе:

- управление приложением (идентификация образа, проверка образа, установка образа, аутентификация);
- средства журналирования;
- регистрация компонентов и сервисов;

- движки выполнения (контейнеры, виртуальные машины);
- языки выполнения (Node.js, Java, Python);
- серверы приложений (Tomcat);
- шины сообщений (RabbitMQ);
- базы данных и архивы (SQL, NoSQL, Cassandra);
- аналитические фреймворки (Spark);
- службы безопасности;
- веб-серверы (Apache);
- инструменты аналитики (Spark, Drool).

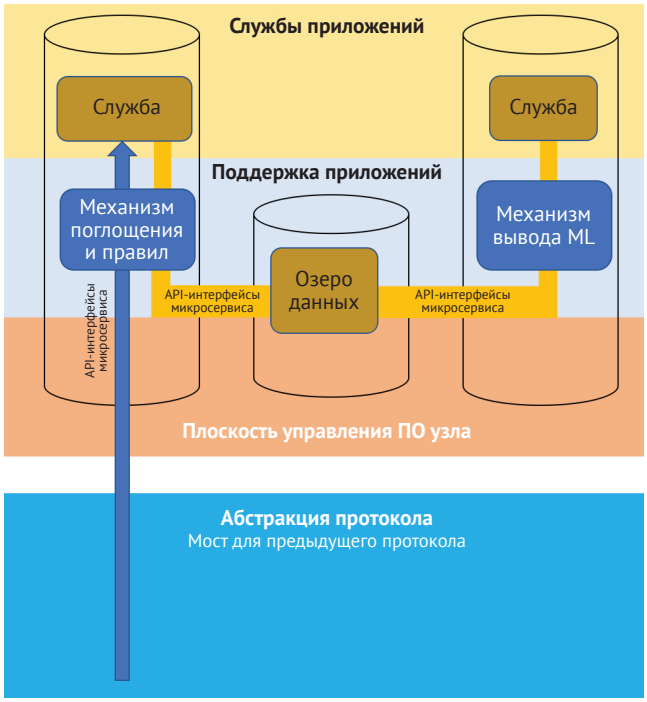


Рис. 10.7 ❖ Пример приложения OpenFog. Здесь могут быть развернуты несколько контейнеров, каждый из которых предоставляет разные службы и функции поддержки

OpenFog предлагает, чтобы эти службы были упакованы в виде контейнеров, как показано на рис. 10.7. Эталонная архитектура не является строгим руководством, и архитектор должен выбрать правильный уровень поддержки, которая может быть включена на ограниченном граничном устройстве. Например, обработка и ресурсы могут допускать только простые правила и запрещать что-либо вроде потокового процессора, не говоря уже о рекуррентной нейронной сети.

Управление узлом и базовое ПО

Это относится к управлению **In-Band (IB)** и определяет, как туманный узел общается с другими узлами в своей области. Узлы также управляются через этот интерфейс для обновлений, статуса и развертывания. Базовое ПО может включать в себя операционную систему узла, пользовательские драйверы и прошивку, протоколы связи и управление, управление файловой системой, программное обеспечение для виртуализации и контейнеризацию для микросервисов.

Этот уровень стека программного обеспечения касается почти любого другого слоя в OpenFog Reference Architecture. Типичными особенностями базового ПО являются:

- **обнаружение сервиса** – позволяет ситуативные модели, доверительные отношения между облаками;
- **обнаружение узла** – позволяет добавлять туманные узлы и присоединять их к кластерам, подобным облачным кластерам;
- **управление состоянием** – позволяет различные вычислительные модели для вычислений с сохранением состояния и без состояния для многих узлов;
- **управление Pub/Sub** – позволяет переносить данные, а не вытягивать их. Кроме того, позволяет использовать уровни абстракции при разработке программного обеспечения.

Эталонная архитектура OpenFog или, в любом случае, как и любая туманная архитектура, должна обеспечивать уровни развертывания. То есть, туманная архитектура не просто ограничена облаком, подключенным к туманному шлюзу, соединенному с несколькими датчиками. На самом деле существует множество топологий, зависящих от масштаба, пропускной способности, нагрузки и экономических факторов, которые могут быть разработаны. Эталонная архитектура должна подходить для множества топологий, так же, как реальное облако может динамически масштабироваться и балансировать нагрузку на основании спроса.

Виртуализация оборудования

Как и обычные облачные системы, OpenFog определяет аппаратное обеспечение как уровень виртуализации. Приложения не должны иметь привязки к определенному набору оборудования. Здесь система должна балансировать нагрузку через туман и перемещаться по ресурсам или добавлять ресурсы по мере необходимости. Все аппаратные компоненты виртуализированы на этом уровне, включая вычисления, сеть и хранилище.

Безопасность узла OpenFog

Консорциум определяет этот уровень как часть безопасности оборудования стека. Туманные узлы более высокого уровня должны иметь возможность контролировать туманные узлы более низкого уровня как часть иерархии в топологии (см. ниже). Одноранговые узлы должны иметь возможность наблюдать

за своими соседями на востоке-западе. Этот слой также выполняет следующие обязанности:

- криптование;
- мониторы слежения и физической безопасности;
- инспекция и мониторинг пакетов (с востока на запад и с севера на юг).

Сеть

Это первый компонент уровня аппаратной системы. Сетевой модуль представляет собой модуль связи восток–запад и север–юг. Сетевой уровень осведомлен о туманной топологии и маршрутизации. Он выполняет физическую функцию маршрутизации к другим узлам. Это большое отличие от традиционной облачной сети, которая виртуализирует все внутренние интерфейсы. Здесь сеть имеет смысл и географическое присутствие в развертывании IoT. Например, родительский узел, управляющий четырьмя другими дочерними узлами, все из которых подключены к камерам, может нести ответственность за объединение видеоданных из четырех источников и объединение (слияние) содержимого изображения вместе для создания поля зрения на 360°. Для этого он должен знать, какой дочерний узел относится к тому или иному направлению, и он не может делать это произвольно или случайно.

Требования к сетевому компоненту:

- устойчивость, если линия связи отказывает. По сути, может потребоваться понять, как перестроить сеть, чтобы сохранить поток данных;
- сетевой уровень также является местом для передачи данных и переупаковки от не-IP-датчиков к IP-протоколам. Примерами этого являются Bluetooth, Z-Wave и проводные датчики;
- обработка отказов;
- связывание с различными коммуникационными сетями (Wi-Fi, проводной, 5G);
- обеспечение типичной сетевой инфраструктуры, необходимой при развертывании предприятия (безопасность, маршрутизация и т. д.).

Ускорители

Другим аспектом OpenFog, чем он отличается от других облачных схем, является понятие услуг ускорителя. Ускорители теперь обычны в виде GPGPU и даже FPGA для предоставления услуг для обработки изображений, машинного обучения, компьютерного зрения и восприятия, обработки сигналов и шифрования/дешифрования. OpenFog предусматривает, что туманные узлы могут снабжаться ресурсами и распределены по мере необходимости. В иерархии можно заставить ферму узлов второго или третьего уровня обеспечить дополнительные вычислительные средства по мере необходимости динамически.

Мы можем даже заставить работать другие формы ускорения в тумане, например:

- узлы, предназначенные для массового хранения в случае, если необходимо создать большое озеро данных;
- узлы, которые включают в себя альтернативные линии связи, такие как спутниковая радиостанция в катастрофическом случае, когда не работает вся наземная связь.

Вычисление

Вычислительная часть стека аналогична вычислительной функциональности уровня Nova в OpenStack. Основные функции:

- выполнение задачи;
- мониторинг и обеспечение ресурсов;
- балансировка нагрузки;
- запросы возможностей.

Хранение

Среда хранения архитектуры поддерживает интерфейс низкого уровня к туманному хранилищу. Типы хранения, с которыми мы сталкивались ранее, такие как озера данных или память рабочего пространства, могут понадобиться на границе для жесткого анализа в реальном времени. Слой хранения также будет управлять всеми традиционными типами устройств хранения, такими как:

- массивы виртуальной памяти;
- заменяемые диски;
- flash-накопители;
- RAID;
- шифрование данных.

Инфраструктура аппаратной платформы

Уровень инфраструктуры – это не столько фактический уровень между программным и аппаратным обеспечением, сколько физическая и механическая структура туманного узла. Поскольку туманные устройства будут находиться в суровых и удаленных местах, они должны быть прочными и устойчивыми, а также автономными. OpenFog определяет случаи, которые необходимо учитывать при развертывании тумана, в том числе:

- размеры, мощность и весовые характеристики;
- системы охлаждения;
- механическое закрепление и удержание;
- механики обслуживания;
- устойчивость к физическому воздействию и отчетность.

Абстракция протокола

Уровень абстракции протокола связывает самые нижние элементы системы IoT (датчики) с другими слоями туманного узла, другими туманными узлами и облаком. OpenFog поддерживает модель абстракции для идентификации и связи с сенсорным устройством через слой абстракции протокола. Абстрагируя интерфейс к датчикам и периферийным устройствам, гетерогенную смесь датчиков

можно развернуть на одном туманном узле, например, аналоговые устройства, которые проходят через цифро-аналоговые преобразователи, а также цифровые датчики. Даже интерфейсы к датчикам могут быть индивидуализированы, например, Bluetooth для температурных устройств в транспортных средствах может подключаться к другим датчикам двигателя, датчикам сопряжения SPI на различных автомобильных электроприборах и датчикам GPIO к различным датчикам открытия двери и случая кражи. Абстрагируя интерфейс, верхние слои стека программного обеспечения могут обращаться к таким разрозненным устройствам с помощью стандартизованного подхода.

Датчики, приводы и системы управления

Это нижний уровень стека IoT: датчики и граничные устройства. Эти устройства могут быть умными, немыми, проводными, беспроводными, располагаться ближе, дальше и т. д. Однако объединение в том, что они каким-то образом сообщаются с туманным узлом, а тот несет ответственность за обеспечение, защиту и управление каждым датчиком.

Amazon Greengrass и лямбда-функции

В этом разделе мы рассмотрим альтернативную туманную службу Amazon Greengrass. Amazon предоставляла в течение многих лет ведущие облачные сервисы и инфраструктуру мирового класса, такие как AWS, S3, EC2, Glacier и другие. С 2016 г. Amazon инвестировала в новый стиль обработки граничных данных под названием Greengrass. Greengrass – это расширение AWS, которое позволяет программисту погружать клиента в туман, шлюз или интеллектуальное сенсорное устройство.

Аналогичным образом другие туманные фреймворки, цель которых состоит в том, чтобы обеспечить решение для сокращения времени ожидания и времени отклика, снизят затраты на пропускную способность и обеспечат безопасность края. Особенности Greengrass следующие:

- данные из кэша в случае потери соединения;
- синхронизация данных и состояния устройства с облаком AWS при повторном подключении;
- локальная безопасность (службы проверки подлинности и авторизации);
- брокер сообщений на устройстве и вне устройства;
- фильтрация данных;
- управление и контроль устройством и данными;
- объединение данных;
- работа в автономном режиме;
- итеративное обучение;
- вызов любой службы AWS непосредственно из Greengrass на крае.

Чтобы использовать Greengrass, программа создаст облачную платформу в AWS IoT и определит определенные лямбда-функции в облаке. Эти функции Lambda затем назначаются граничным устройствам и развертываются на тех

устройствах, на которых запущен клиент и которые уполномочены выполнять Greengrass Lambdas. В настоящее время функции Lambda написаны на Python 2.7. Shadows – это абстракции JSON в Greengrass, которые представляют состояние устройства и функции Lambda. При необходимости они синхронизируются с AWS.

За кулисами связь между Greengrass на краю и AWS в облаке осуществляется через MQTT.

Обратите внимание, что функции Lambda не следует путать с архитектурой Lambda, упомянутой ранее. Лямбда-функция в контексте Greengrass относится к управляемой событием функции вычислений.

Пример Lambda-определения, используемый в Greengrass, будет выглядеть следующим образом. С консоли в AWS мы запускаем командную строку; мы запускаем следующий инструмент и определяем функцию Lambda по имени:

```
aws greengrass create-function-definition --name “sensorDefinition”
```

В результате получим следующее:

```
{
 "LastUpdatedTimestamp": "2017-07-08T20:16:31.101Z",
 "CreationTimestamp": "2017-07-08T20:16:31.101Z",
 “Id”: “26309147-58a1-490e-a1a6-0d4894d6ca1e”,
 "Arn":"arn:aws:greengrass:us-west-
2:123451234510:/greengrass/definition/functions/26309147-58a1-490e-a1a6-0d4894d6ca1e",
 "Name": "sensorDefinition"
}
```

Теперь мы создаем объект JSON с определениями функции Lambda, используем указанный выше идентификатор и вызываем функцию `create-functiondefinition-version` из командной строки:

- `Executable` – исполняемая функция Lambda по имени;
- `MemorySize` – это объем памяти, выделяемый обработчику;
- `Timeout` – время ожидания в секундах до истечения времени ожидания таймера.

Ниже приведен пример объекта JSON для использования с функцией Lambda:

```
aws greengrass create-function-definition-version --function-definition-id
"26309147-58a1-490e-a1a6-0d4894d6ca1e". --functions
'[
{
    "Id": "26309147-58a1-490e-a1a6-0d4894d6ca1e",
    "FunctionArn": "arn:aws:greengrass:us-west-
2:123451234510:/greengrass/definition/functions/26309147-58a1-490e-a1a6-
0d4894d6ca1e",
    "FunctionConfiguration": {
        "Executable": "sensorLambda.sensor_handler",
        "MemorySize": 32000,
        "Timeout": 3
    }
}]'
```

Для обеспечения и создания подписки между граничным узлом и облаком необходимы еще несколько шагов, но будет все равно развернут обработчик Lambda. Это допускает альтернативный подход к туманным вычислениям, как это предусмотрено в Amazon. Эту модель можно рассматривать как способ расширения облачных сервисов для граничного узла и край, обладающий полномочиями для вызова любого ресурса, что рассматривается как предоставляемое облако. Это по определению настоящая платформа туманных вычислений.

Туманные топологии

Туманные топологии могут существовать во многих формах, и архитектору необходимо учитывать несколько аспектов при разработке туманной системы «из конца в конец». В частности, при разработке топологии вступают в силу ограничения, такие как стоимость, нагрузка на процессор, интерфейс производителя и передача между востоком-западом. Туманная сеть может быть простой, как граничный маршрутизатор с поддержкой тумана, соединяющий датчики с облачным сервисом. Она также может усложняться в многоуровневой туманной иерархии тумана с разной степенью способности обработки в каждом слое и ролями на каждом уровне, одновременно перераспределяя нагрузки обработки, когда и где это необходимо (с востока на запад и с севера на юг). Определяющие факторы моделей основаны на следующем:

- **уменьшение объема данных** – например, система собирает неструктурированные видеоданные с тысяч датчиков или камер, агрегирует данные и ищет конкретные события в режиме реального времени. Если это так, то сокращение набора данных будет значительным, так как тысячи камер будут ежедневно производить сотни ГБ данных, а туманным узлам нужно будет перевести большие объемы данных в простые формы: «да», «нет», «опасность», «токены безопасности события»;
- **количество граничных устройств** – если система IoT представляет из себя всего лишь один датчик, то набор данных мал и может не оправдывать использование граничного туманного узла. Однако, если число датчиков растет или, в худшем случае, рост количества датчиков непредсказуем и динамичен, то туманная топология может потребовать масштабирования вверх или вниз динамически. Мы рассматриваем как пример стадион с использованием Bluetooth-маяка. Поскольку аудитория растет на определенных площадках, система должна иметь возможность масштабироваться нелинейно. В других случаях стадион может занимать лишь небольшую площадь, и ему нужны только ограниченные ресурсы обработки и подключения;
- **возможности туманного узла** – в зависимости от топологии и стоимости некоторые узлы могут лучше подходить для подключения к системам WPAN, тогда как другие узлы в иерархии могут иметь дополнительные возможности по обработке для машинного обучения, распознавания образов или обработки изображений. Примером могут быть граничные

туманные узлы, которые управляют безопасной сеткой Zigbee и имеют специальное оборудование для аварийных ситуаций или безопасности WPAN. Выше этого уровня будет существовать узел обработки тумана, который будет иметь дополнительную оперативную память и GPGPU для поддержки потоков необработанных данных из шлюзов WPAN;

- **надежность системы** – архитектору может потребоваться рассмотрение форм отказа в модели IoT. Если один крайний туманный узел дал сбой, другой мог бы занять его место для выполнения какого-либо действия или сервиса. Этот случай важен в жизненно-критических случаях или при работе в реальном времени. Таким же образом дополнительные туманные узлы могут подключаться по требованию; избыточные узлы могут потребоваться в ситуациях обеспечения отказоустойчивости. В случае отсутствия дополнительных избыточных узлов некоторая обработка может совместно производиться соседними узлами за счет использования системных ресурсов и задержки, но система будет продолжать функционировать. Оконечный вариант использования – это то, где соседние узлы действуют как сторожевые псы друг для друга. В случае сбоя туманного узла или сбоя связи с узлом сторожевой таймер сигнализирует о событии сбоя для облака и может выполнять локальные критические действия. Хорошим примером является случай, когда туманный узел сбоит при контроле трафика на шоссе; соседний узел может увидеть отказ точки, предупредить облако о событии и подать сигнал на щит на шоссе, чтобы уменьшить скорость.

Простейшим решением для тумана является блок обработки на краю (шлюз, тонкие клиенты, маршрутизатор), расположенные рядом с матрицей датчиков. Здесь туманный узел может использоваться в качестве шлюза для сети или mesh-сети WPAN и обмениваться данными с хостом (рис. 10.8).

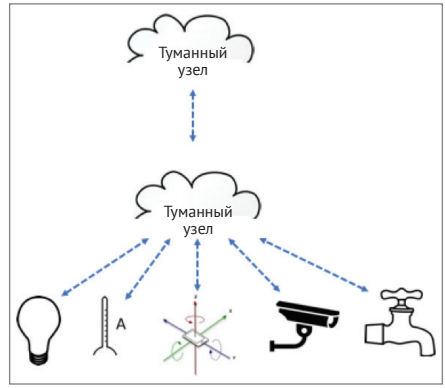


Рис. 10.8 ❖ Простая туманная топология.
Гранично-туманное устройство управляет массивом датчиков
и может взаимодействовать с другим туманным узлом на основе M2M

Следующая базовая туманная топология включает облако в качестве родителя по туманной сети. Туманный узел в этом случае будет собирать данные, защищать край и выполнять обработку, необходимую для связи с облаком. Эту модель отделяет от граничных вычислений то, что сервисные и программные уровни туманного узла разделяют отношения с облачным фреймворком (рис. 10.9).

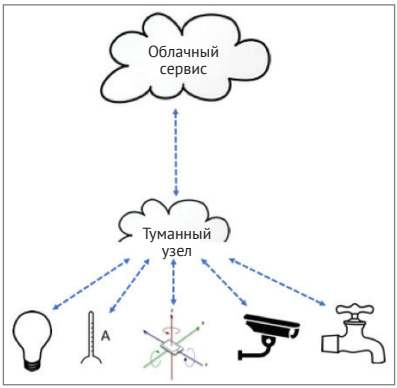


Рис. 10.9 ❖ Туман к облачной топологии. Здесь туманный узел устанавливает отношения с поставщиком облака

Следующая модель использует несколько туманных узлов, ответственных за услуги и обработку на краю, и каждый из них подключается к набору датчиков. Родительское облако обеспечивает каждый туманный узел, как если бы он был единственным узлом. Каждый узел имеет уникальный идентификатор, чтобы обеспечить уникальный набор сервисов на основе расположения. Например, каждый туманный узел может находиться в другом месте для розничной сети. Туманные узлы также могут связывать и передавать данные с востока на запад между граничными узлами. Примером могут служить условия холодного хранения, когда необходимо поддерживать и управлять вентиляторами и морозильниками, чтобы предотвратить порчу продуктов питания. У розничного продавца может быть несколько вентиляторов в разных местах, все управляются единой облачной службой, но работают с туманными узлами на краю (рис. 10.10).

Следующая модель расширяет вторую топологию, добавляя возможность надежно и конфиденциально общаться с несколькими поставщиками облаков от нескольких туманных узлов. В этой модели могут быть развернуты несколько родительских облаков. Например, в умных городах могут существовать многочисленные географические районы, и они могут охватываться различными муниципалитетами. Каждый муниципалитет может выбрать одного облачного провайдера, сравнив его с другими, но все муниципалитеты управля-

ются с использованием одного утвержденного и бюджетного производителя камер и датчиков. В этом случае производитель камер и датчиков будет иметь один экземпляр облака, сосуществующий с несколькими муниципалитетами. Туманные узлы должны иметь возможность доставлять данные нескольким поставщикам облачных вычислений (рис. 10.11).

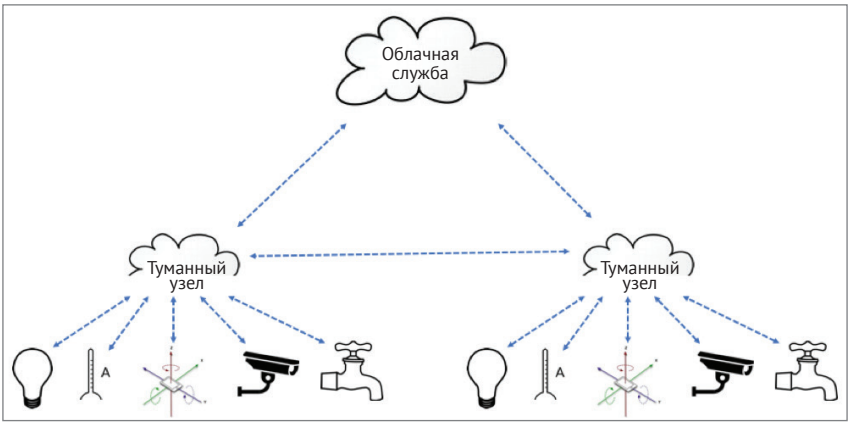


Рис. 10.10 ❖ Несколько туманных узлов с одним основным облаком

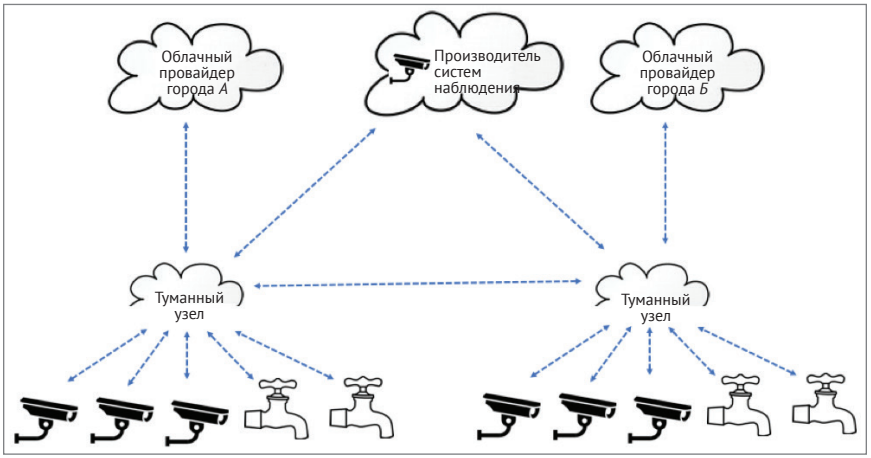


Рис. 10.11 ❖ Несколько туманных узлов с несколькими облачными провайдерами. Облака могут представлять собой смесь общественных и частных облаков

Туманные узлы также не нуждаются в индивидуальном соединении, связывающем датчики с облаками «один к одному». Туманные узлы могут быть помещены в стек, многоуровневыми или даже законсервированными до тех пор,

пока не понадобятся. Иерархия уровней туманных узлов друг над другом может казаться противоречивой, если мы пытаемся уменьшить задержки, но, как упоминалось ранее, узлы могут быть специализированными. Например, узлы, расположенные ближе к датчикам, могут предоставлять услуги в режиме реального времени или иметь ограничения по стоимости, требующие от них минимального объема данных для хранения и вычисления. Уровни над ними могут предоставлять вычислительные ресурсы, необходимые для агрегированного хранения, машинного обучения или распознавания изображений с использованием дополнительных запоминающих устройств или GPGPU-процессоров. Следующий пример иллюстрирует использование на примере освещения города.

Здесь несколько камер чувствуют движение пешеходов и трафик; туманные узлы, наиболее близкие к камерам, выполняют агрегацию и извлечение признаков и передают эти данные вверх до следующего уровня. Родительский туманный узел извлекает данные и выполняет необходимое распознавание изображений с помощью алгоритма глубокого обучения. Если будет наблюдаться интересное событие (например, пешеходная прогулка ночью вдоль пути), событие будет отправлено в облако. Компонент облака регистрирует событие и сигнализирует множеству уличных фонарей в окрестности пешехода, чтобы они увеличили освещенность. Эта картина будет продолжаться до тех пор, пока туманные узлы видят движение пешехода. Конечной целью является общая экономия энергии, чтобы каждая уличная лампочка не работала на полной интенсивности все время (рис. 10.12).

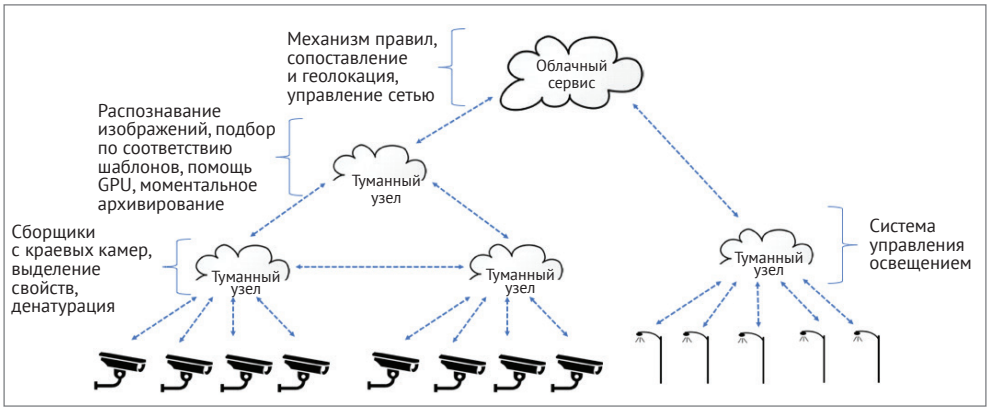


Рис. 10.12 ❖ Многоуровневая туманная топология. Туманные узлы складываются в стек в иерархии уровней, чтобы обеспечить дополнительные услуги или абстракции

Заключение

Анализ данных и получение значимых выводов из сенсора является целью IoT. Когда мы масштабируем до тысяч, миллионов и потенциально миллиар-

дов объектов, которые передают и передают данные без остановок, мы должны внедрять передовые инструменты для получения, хранения, передачи, анализа и прогнозирования значений из этого моря данных. Облачные вычисления – один из элементов, позволяющих использовать эту услугу в виде кластеров масштабируемого оборудования и программного обеспечения. Туманные вычисления делают облачную обработку ближе к краю для решения проблем с задержкой, безопасностью и расходами на связь. Обе технологии работают вместе, чтобы обеспечить работу аналитических пакетов в виде движков правил для сложных агентов обработки событий. Выбор модели облачных провайдеров, фреймворков, туманных узлов и модулей аналитики является важной задачей, и множество материалов посвящено глубокому рассмотрению семантики программирования и создания этих сервисов. Архитектор должен понять топологию и конечную цель системы, чтобы построить структуру, которая удовлетворяет сегодняшние потребности и масштабируется в будущем.

В следующей главе мы обсудим часть аналитики данных IoT. Облако, безусловно, может содержать ряд аналитических функций, однако мы должны быть готовы понять, что определенный анализ должен выполняться на краю, более близком к источнику данных (датчику), или, если это имеет смысл, в облаке (используя долгосрочные исторические данные).

Глава 11

Анализ данных и машинное обучение в облачных и туманных платформах

Основная ценность IoT-систем состоит не в архивации миллионов событий, сгенерированных датчиками, а в их интерпретации и принятии соответствующих решений. Мир, в котором миллиарды сущностей соединяются и взаимодействуют друг с другом и с облачными платформами, захватывает дух, но нас в первую очередь интересуют данные, их содержимое, то, что в них не попало, и закономерности, которые из них можно вывести. Это та область интернета вещей, относящаяся к науке о данных и их анализу, является, наверное, наиболее ценной для клиентов.

В интернете вещей анализируется такая информация:

- структурированные данные (SQL-хранилище) предсказуемого формата;
- неструктурированные данные (необработанное видео или сигналы) высокой степени случайности и вариативности;
- частично структурированные данные (Twitter-потоки) с некоторой степенью случайности вариативности.

Иногда данные необходимо интерпретировать и анализировать в режиме реального времени в виде потока, а иногда они архивируются с последующим углубленным анализом в облаке. Это стадия потребления данных. В некоторых случаях информацию приходится сопоставлять с другими источниками на лету, а иногда она просто записывается и сбрасывается в так называемое *озеро данных* (англ. *data lake*), вроде Hadoop.

Дальше идет этап перемещения. Такие системы обмена сообщениями, как Kafka, направляют данные в поточный или пакетный процессор (или в оба сразу). Данные обрабатываются в виде непрерывного потока. Этот процесс обычно имеет определенные ограничения и высокую производительность, так как информация обрабатывается в памяти. В связи с этим обработка данных должна происходить не медленней, чем их поступление. И если в облаке

можно рассчитывать на работу в режиме, близком к реальному времени, то промышленные устройства или беспилотные автомобили не дают жестких гарантий относительно производительности.

С другой стороны, пакетная обработка хорошо подходит для больших объемов данных, особенно при сопоставлении показателей датчиков с ранее сохраненной информацией.

Следующий этап может включать в себя прогнозирование и возвращение ответа. Данные могут быть выведены на панель управления, записаны в журнал или возвращены обратно пограничному устройству, которое может принять определенные меры для решения некоторых проблем.

Эта глава посвящена различным моделям анализа данных: от обработки сложных событий до машинного обучения. Вам будет предложено несколько сценариев, на примере которых вы узнаете, в каких ситуациях подходит та или иная модель.

Простой анализ данных в интернете вещей

Анализ данных занимается поиском событий, как правило, в потоках информации. Существует несколько видов событий и ролей, которые должно поддерживать устройство поточного анализа в режиме реального времени. Ниже приведены общие категории аналитических функций, основанные на докладе Срината Переры и Срискандараи Сухотаяна «Шаблоны решений для поточного анализа в режиме реального времени», представленном на 9-й Международной конференции по распределенным событийным системам (DEBS ’15) в Нью-Йорке:

- **предварительная обработка** – отбрасывание малоинтересных событий, денатурирование, извлечение свойств, сегментация, перевод данных в более подходящий формат (хотя озера данных предпочитают избегать немедленного преобразования), добавление таких атрибутов, как ярлыки (озерам данных ярлыки не нужны);
- **оповещение** – проверка данных; если они не соответствуют каким-то граничным условиям, отправляется предупреждение. Элементарный пример: температура превышает определенный лимит, установленный в датчике;
- **окна событий** – создается подвижный диапазон (окно), в котором действуют свои собственные правила. Окно может быть ограничено по времени (например, на протяжении одного часа) или длине (например, в рамках 2000 показаний датчика). Окна могут быть подвижными (например, проверка последних 10 событий датчика и возвращение результатов при каждом новом событии) и пакетными (когда событие генерируется только по завершении окна). Эта функция хорошо подходит для создания правил и событий, занимающихся подсчетом. К примеру, вы можете узнать количество скачков температуры за последний час и сделать вывод о том, что одно из устройств имеет дефект;

- **соединения** – объединяют несколько потоков данных в один. Можно привести пример в сфере логистики. Допустим, компания рассылает посылки и отслеживает их с помощью специальных маяков, а ее многочисленные грузовики, самолеты и объекты недвижимости шлют потоки геолокационных данных. Таким образом, мы имеем два потока: один для посылки, а другой для заданного грузовика. Когда грузовик подбирает посылку, эти два потока объединяются;
- **ошибки** – работая с миллионами датчиков, вы непременно столкнетесь с потерей данных, их искажением или неправильным порядком следования. Это актуально для интернета вещей с его множественными асинхронными и независимыми потоками. Например, информация может затеряться в сотовой сети, если автомобиль заедет в подземный гараж. Этот аналитический шаблон сопоставляет данные в рамках одного потока, пытаясь найти подобные ошибки;
- **базы данных** – аналитический пакет должен взаимодействовать с каким-то хранилищем данных. Например, если данные поступают из ряда датчиков, таких как ярлыки Bluetooth, сигнализирующие о хищении или пропаже устройства, вам нужно будет сопоставлять их с базой данных идентификаторов разыскиваемых ярлыков;
- **временные события и шаблоны** – эта функция чаще всего применяется в сочетании с окнами событий, упомянутыми выше. Здесь целевым шаблоном являются наборы или последовательности событий. Это такой конечный автомат. Представьте, что мы отслеживаем состояние устройства по его температуре, вибрациям и шуму, который оно издает. Последовательность временных событий может выглядеть так:
  1) определить, превышает ли температура 100 °C;
  2) определить, превышает ли уровень вибраций 1 м/с;
  3) определить, издает ли устройство звук громкостью 110 дБ;
  4) отправить оповещение, если все эти события происходят в такой последовательности;
- **отслеживание** – событие генерируется по факту наличия или отсутствия каких-либо данных в заданном месте или в определенный момент времени. Простейшим примером является геолокация для служебных автомобилей, когда компании необходимо знать их точное местоположение и когда они в последний раз посещали то или иное место. Эту функцию можно применять в сельском хозяйстве, коммунальном вывозе мусора, уборке снега, для отслеживания пассажиропотоков, пациентов, ценных активов, багажа и т. д.;
- **тенденции** – этот шаблон особенно полезен в прогностическом облуживании. Он подразумевает создание правила для обнаружения событий на основе наборов данных, связанных по времени. Временные события имеют похожий принцип работы, но вместо времени в них фигурирует понятие последовательности. В этой модели время является одним из измерений процесса. Актуальная история событий, связанных по вре-

мени, может использоваться сельском хозяйстве. Например, к голове коровы можно прикрепить датчик, который будет отслеживать ее перемещения и температуру. Чтобы посмотреть, двигалась ли корова на протяжении дня, можно составить последовательность событий. Отсутствие движения может указывать на то, что животное заболело или умерло;

- **пакетные запросы** – пакетная обработка обычно оказывается более комплексной и глубокой, чем вычисления в режиме реального времени. Хорошо спроектированная поточная платформа может разделять анализ на несколько частей и передавать их системе пакетной обработки. Мы еще вернемся к этой теме при обсуждении lambda-функций;
- **глубокий анализ** – при обработке в режиме реального времени решения о возникновении событий принимаются на лету. Должно ли это событие сгенерировать уведомление – это вопрос, ответ на который может потребовать дополнительной обработки, которая выполняется позже. Такой подход работает, поскольку события подобного рода должны быть редкими; и, пока новые события создаются в режиме реального времени, старые передаются в движок глубокой обработки. В качестве примера можно привести систему видеонаблюдения. Представьте, что платформа умного города бьет тревогу, если пропал ребенок, и запускает простую модель распознавания и классификации изображений для поточных движков реального времени. Модель способна обнаружить номерной знак автомобиля, в котором находится ребенок, или даже логотип на детской футболке. Первым делом нужно будет захватить изображения с номерными знаками и логотипами на одежде прохожих и передать их в облако. Из миллионов полученных образцов система анализа попытается распознать нужный номер или логотип. Это первый этап. Затем, чтобы исключить ложные срабатывания, распознанный кадр (вместе с соседними кадрами) будет направлен в более продвинутую систему анализа, которая использует более подробные алгоритмы распознавания объектов (слияние изображений, увеличение разрешения, машинное обучение);
- **модели и обучение** – модель первого уровня, описанная выше, на самом деле может являться механизмом логического вывода для системы машинного обучения. Подобные системы строятся на обучаемых моделях и могут использоваться на лету при анализе в режиме реального времени;
- **обмен сигналами** – действия часто приходится возвращать обратно пограничному компоненту или датчику. Типичным примером является автоматизация и безопасность фабрики. Если температура устройства превысит определенный лимит, это событие нужно не только записать в журнал, но и отправить пограничному компоненту, чтобы замедлить устройство. Механизм коммуникаций в системе должен быть двунаправленным;
- **управление** – наконец, у нас должна быть возможность управлять всеми этими средствами анализа. Для работы с системой должны быть предус-

мотрены такие функции как запуск, остановка, создание отчетов, запись в журнал и отладка.

Теперь мы сосредоточимся на построении облачной аналитической архитектуры, способной потреблять непредсказуемые и непрерывные потоки данных и интерпретировать их в режиме, максимально близком к реальному времени.

Верхний уровень облачной архитектуры

На рис. 11.1 показан типичный путь следования данных от датчика к панели управления. Данные проходят через несколько промежуточных звеньев (узлы WPAN, широкополосный канал, облачное хранилище в виде озера данных и т. д.). При выборе архитектуры для построения облачного аналитического решения следует учитывать эффект масштабирования. Решения, принятые на ранних этапах проектирования, могут сгодиться для десятка IoT-узлов и одного облачного кластера, но, когда число конечных IoT-устройств вырастет до тысяч, а система начнет охватывать несколько географических зон, масштабирование может оказаться неэффективным.

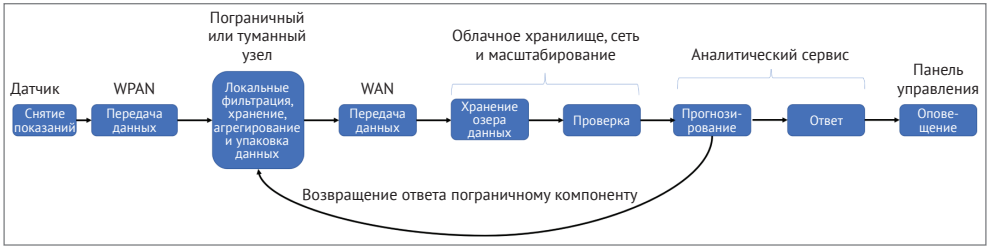


Рис. 11.1 ❖ Типичная схема передачи данных от датчика к облаку

Аналитические функции облачной системы (прогнозирование-ответ) могут принимать несколько форм:

- **система правил –** определяет действие и возвращает результат;
- **поточная обработка** – здесь события, такие как показания датчиков, передаются поточному процессору. Путь обработки представляет собой граф с операторами в качестве узлов; при этом данные переходят от одного оператора к другому. Каждый узел содержит код для определенного этапа обработки и ссылку на следующий узел в графе. Такой граф можно реплицировать и выполнять параллельно в кластере, поэтому его можно масштабировать на сотни компьютеров;
- **обработка сложных событий** – эта часть системы основана на языке запросов высокого уровня SQL и занимается обработкой событий. Она заточена под низкие задержки;
- **Lambda-архитектура** – эта модель пытается сбалансировать пропускную способность системы и уровень задержек, выполняя пакетную обработку и параллельные вычисления с громадными наборами данных.

Причина, по которой мы обсуждаем анализ в режиме реального времени, связана с тем, что данные одновременно поступают из миллионов узлов; это происходит непрерывно и асинхронно, с различными ошибками, проблемами форматирования и нестыковками по времени. В Нью-Йорке установлено 250 000 уличных фонарей (www.nyc.gov/html/dot/html/infrastructure/streetlights.shtml). Представьте, что каждый фонарь является умным – то есть, он регулирует свою яркость в зависимости от движения вокруг себя, и, если поблизости никого нет, он остается затемненным и экономит электроэнергию (на это уходит 2 байта). Каждый фонарь может отслеживать фонари, которые требуют починки (1 байт). Кроме того, он измеряет температуру (1 байт) и влажность (1 байт), помогая генерировать локальный прогноз погоды. Наконец, данные, с которыми он работает, содержат его идентификатор и временную метку (8 байт). В штатном режиме фонари генерируют, в общей сложности, 250 000 сообщений в секунду, хотя в часы пик и на праздники (а также из-за большого скопления людей и благодаря туристическим достопримечательностям) этот показатель может достигать 325 000. Допустим, наш облачный сервис способен обрабатывать 250 000 сообщений в секунду – то есть, отставание может достигать 75 000 событий в секунду. Если час пик действительно затянется ровно на 1 час, за это время у нас накопится 270 000 000 необработанных событий. Чтобы дать системе шанс наверстать упущенное, нам придется увеличить вычислительную мощность кластера или уменьшить входящий поток. Если каждую секунду в периоды низкой активности убирать из потока 200 000 сообщений, облачному кластеру понадобится 1,1 часа и 585 Мб памяти (270 миллионов событий по 13 байт каждое), чтобы нивелировать отставание.

Мощность облачной системы можно выразить следующими уравнениями:

$$C = \text{Мощность кластера}\left(\frac{\text{событие}}{c}\right);$$

$R_{\text{событие}}$ = Частота поступления событий;

$T_{\text{всплеск}}$ = Период всплеска событий;

T_c = Время на устранение отставания;

$M_{\text{отставание}}$ = Отстающие сообщения (размер);

$$\text{Отставание} = \begin{cases} 0, \text{ где } R_{\text{событие}} \le C \\ R_{\text{событие}} - C, \text{ где } R_{\text{событие}} > C \end{cases};$$

$$M_{\text{отставание}} = \text{Отставание} \times M_{\text{размер}}$$

$$T_c = \frac{(R_{\text{событие}} \times T_{\text{всплеск}}) + M_{\text{отставание}}}{c}.$$

Система правил

Система правил – это программный компонент, который выполняет какие-то действия в ответ на события. Например, если влажность в комнате превысит 50%, собственнику отправляется SMS. Этот механизм также называют *си-*

стемой управления бизнес-правилами (англ. *Business Rule Management System,* или *BRMS*).

Некоторые системы правил обладают состоянием. Это означает, что они могут хранить историю событий и выполнять разные действия в зависимости от их порядка, количества или характера их поступления. Системы, не имеющие состояния, проверяют только текущее событие (рис. 11.2).

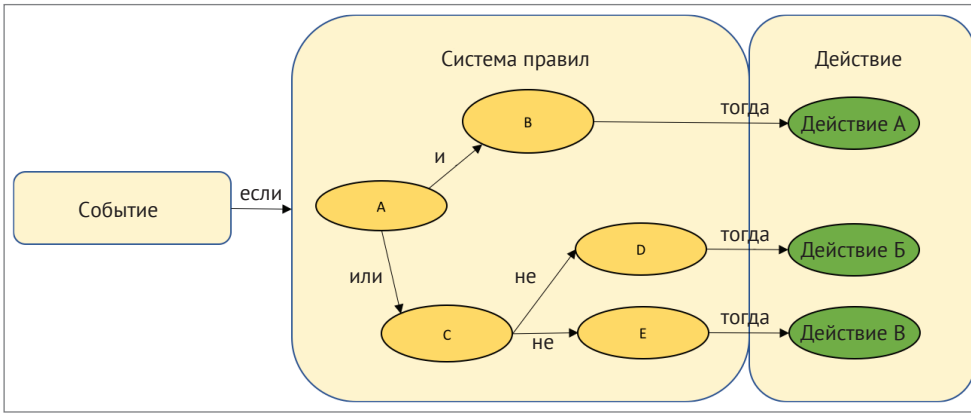


Рис. 11.2 ❖ Пример простой системы правил

В нашем примере роль системы правил будет выполнять Drools. Drools – это BRMS, разработанная компанией Red Hat и распространяемая под лицензией Apache 2.0. Ее промышленная версия называется JBoss Enterprise. Все свои объекты Drools хранит в *рабочей памяти*; это такой набор событий, присланных датчиками, которые следует сравнить для удовлетворения заданного правила. Drools поддерживает два вида сцепления: прямое и обратное. Сцепление – это метод логического вывода, позаимствованный из теории игр.

При прямом сцеплении данные принимаются до тех пор, пока не будет сформирована цепочка правил. Последняя может представлять собой последовательность операторов `if/then`, как на рис. 11.2. Прямое сцепление непрерывно ищет данные, удовлетворяющие одному из маршрутов `if/then`, чтобы завершить действие. Обратное сцепление двигается в противоположном направлении: от действия к данным, а не наоборот. Ниже представлен псевдокод, демонстрирующий простую систему правил:

```
Smoke Sensor = Smoke Detected
Heat Sensor = Heat Detected

if (Smoke_Sensor == Smoke_Detected) && (Heat_Sensor == Heat_Detected) then Fire
if (Smoke_Sensor == !Smoke_Detected) && (Heat_Sensor == Heat_Detected) thenFurnace_On
if (Smoke_Sensor == Smoke_Detected) && (Heat_Sensor == !Heat_Detected) then Smoking
if (Fire) then Alarm
if (Furnace_On) then Log_Temperature
if (Smoking) then SMS_No_Smoking_Allowed
```

Представьте, что:

- Smoke_Sensor (датчик задымленности) выключен;
- Heat_Sensor (датчик тепла) включен.

Прямое сцепление остановилось бы на втором условии, делая вывод о том, что температура записывается в журнал.

Обратное сцепление пытается доказать, что отопление включено. Для этого оно двигается в противоположном направлении:

1) можем ли мы доказать, что температура записывается в журнал? Взгляните на следующий код:

```
if (Furnace_On) then Log_Temperature
```

2) так как температура записывается в журнал, условие (Furnace_On) становится новой целью:

```
if (Smoke_Sensor == !Smoke_Detected) && (Heat_Sensor == Heat_Detected) then Furnace_On
```

3) мы уже доказали, что отопление включено, поэтому новое условие состоит из двух частей: Smoke_Sensor и Heat_Sensor. Система правил разбивает его на две цели:

```
Smoke_Sensoroff
Heat_Sensoron
```

4) теперь система правил пытается достичь обеих целей. Во время этого процесса завершается логический вывод.

Прямое сцепление позволяет реагировать на новые данные по мере их поступления, что может привести к новым логическим выводам.

Семантический язык Drools был специально упрощен. Он состоит из таких базовых элементов:

- сессии, которые определяют правила по умолчанию;
- точки входа, определяющие правила, которые будут использоваться;
- условные выражения (when);
- предпринимаемые действия (then).

В следующем псевдокоде представлено простое правило в Drools. Операция insert выполняет изменение рабочей памяти. Обычно это делается в случае, когда условие оказывается истинным.

```
rule "Furnace_On"
when
Smoke_Sensor(value &gt; 0) &&Heat_Sensor(value &gt; 0)
then
insert(Furnace_On())
end
```

После выполнения всех правил программа может сделать запрос к рабочей памяти, чтобы узнать, какие условия оказались истинными. Для этого используется следующий синтаксис:

```
query "Check_Furnace_On"
$result: Furnace_On()
end
```

Данное правило имеет два шаблона:

- **синтаксический**: формат данных, соответствие, хеш, диапазон значений;
- **семантический:** значение должно входить в список, а счетчик значений с высокой температурой не должен превысить 20 за один час. В сущности, это значимые события.

Drools поддерживает создание очень сложных и подробных правил, для хранения которых может даже понадобиться отдельная база данных. Семантика языка предусматривает шаблоны, вхождение в диапазон, подсчет количества срабатываний правила, сопоставление типов и работу с коллекциями объектов.

Потребление информации: потоки, обработка и озера данных

IoT-устройство обычно соединено с каким-то датчиком или другим устройством, которое измеряет или отслеживает условия в реальном мире. Это делается асинхронно по отношению к остальным технологиям интернета вещей. То есть, датчик пытается транслировать данные вне зависимости от того, слушает ли его облачный или туманный узел. Это важно, поскольку во многих организациях данные представляют наибольшую ценность. Даже если большая часть информации оказывается избыточной, в какой-то момент может произойти важное событие. В связи с этим данные передаются в виде потоков.

Поток информации, передающийся из датчика в облако, воспринимается как:

- постоянный и бесконечный;
- асинхронный;
- неструктурированный или структурированный;
- максимально близкий к режиму реального времени.

Ранее, в главе 10, мы уже обсуждали временные задержки, свойственные облакам. Мы также узнали, что туманные вычисления помогают решить эту проблему. Но и без туманных узлов предпринимаются усилия для оптимизации облачной архитектуры, чтобы обеспечить поддержку режима реального времени в интернете вещей. Для этого облако должно уметь работать с непрерывным потоком данных. Альтернативой является *пакетная обработка*. Большинство аппаратных платформ используют одни и те же методы работы с потоками, перемещая данные из одного блока в другой; при этом на каждом этапе их поступление активирует следующую функцию. Важным элементом снижения общей латентности является тщательное использование хранилища данных и доступ к файловой системе.

В связи с этим большинство поточных фреймворков выполняют свои операции в памяти, пытаясь полностью исключить временное хранение инфор-

мации в файловой системе. Важность этого подхода подчеркивается в статье Майкла Стоунбрейкера, Угура Кетинтемела и Стэна Здоника «8 требований к обработке потоков в режиме реального времени» (SIGMOD Rec. 34, 4, 2005 г.). Этому шаблону способствует хорошо спроектированная очередь сообщений. Построение успешной облачной архитектуры, способной масштабироваться на сотни и миллионы узлов, требует тщательного планирования.

К тому же, потоки данных редко бывают идеальными. В ситуации, когда сотни тысяч датчиков асинхронно передают свои показания, некоторые данные будут теряться (при потере связи с датчиком), иметь некорректный формат (ошибка при передаче) или поступать не в том порядке (если облако принимает информацию из нескольких источников). Поточная система должна как минимум:

- масштабироваться при росте и всплесках количества событий;
- предоставлять возможность публикации/подписки в своем API-интерфейсе;
- стремиться к минимизации задержек;
- обеспечивать масштабирование правил обработки;
- поддерживать озера и хранилища данных.

Фонд Apache предлагает несколько открытых проектов (с лицензией Apache 2), которые должны помочь в построении архитектуры для обработки потоков. Один из них, Apache Spark, обрабатывает данные в виде небольших пакетов. Это крайне удобно, когда объем памяти в облачном кластере ограничен (например, < 1 Тб). Фреймворк Spark выполняет свои операции в памяти, что снижает латентность и зависимость от файловой системы (как уже упоминалось ранее). Пакетная обработка данных также хорошо подходит при работе с моделями машинного обучения, о которых мы поговорим позже в этой главе. Поддержка пакетных данных реализована в нескольких моделях, таких как сверточная нейронная сеть (англ. *Convolutional Neural Network*). Еще одной альтернативой от Apache является проект Storm, который пытается максимально приблизить облачные вычисления к режиму реального времени. В отличие от Spark он имеет низкоуровневый API-интерфейс и обрабатывает данные в виде крупных наборов событий, не разбивая их на пакеты. Это позволяет добиться низкой латентности (доли секунды).

Чтобы скармливать данные фреймворкам для обработки потоков, можно использовать Apache Kafka или Flume. Apache Kafka – это очередь MQTT, которая принимает показания из различных IoT-датчиков и клиентов, и затем передает их в Spark или Storm. MQTT не занимается буферизацией, поэтому, если с облаком взаимодействуют тысячи клиентов, нам понадобиться некая система, которая будет накапливать сообщения из входящего потока. Это позволит Kafka масштабироваться в зависимости от нагрузки (еще одно важное качество облака) и должным образом реагировать на всплески событий. Kafka поддерживает потоки на скорости 1 00 000 сообщений в секунду. С другой стороны, у нас есть распределенная система Flume, которая собирает, агрегирует

и перемещает данные из одного источника в другой; ее проще использовать *из коробки*. Она также имеет тесную интеграцию с Hadoop. Flum масштабируется не так хорошо, как Kafka, поскольку добавление новых потребителей требует изменения архитектуры системы. Оба фреймворка способны хранить потоки в памяти, не сбрасывая их на диск; но обычно это не самый удачный подход, так как показания датчиков, которые, напомним, вещают одновременно, лучше сохранять в виде, как можно более близком к оригиналу.

Если наша IoT-система состоит из тысяч или миллионов датчиков и конечных узлов, в облачной среде имеет смысл использовать озеро данных. *Озеро данных* – это, в сущности, огромное хранилище, в которое стекается необработанная и неотфильтрованная информация из множества источников. В отличие от обычных файловых систем оно совершенно плоское и не поддерживает иерархии, тома, директории, файлы и папки. Для структурирования содержимого к каждой записи прикрепляется элемент метаданных (ярлык). Классическим примером озера данных является Apache Hadoop, хотя практически все облачные провайдеры используют эту модель в том или ином виде.

Озеро данных отлично подходит для интернета вещей, позволяя хранить информацию независимо от того, структурирована она или нет. Все данные в нем считаются ценными и хранятся на постоянной основе. Такое скопление информации является оптимальным для систем анализа данных. Качество работы алгоритмов зависит от объема данных, которые они потребляют или используют для обучения своих моделей.

На рис. 11.3 представлена концептуальная архитектура на основе традиционной пакетной и поточной обработки. Озеро данных наполняется с помощью экземпляра Kafka. Облако Kafka предоставляет пакетный интерфейс к Spark и передает данные в хранилище.

Компоненты используют стандартные соединения, поэтому топологию данной архитектуры можно модифицировать несколькими способами.

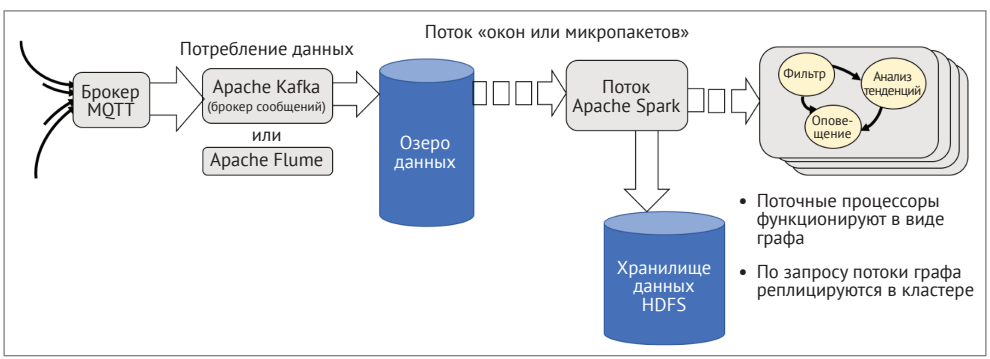


Рис. 11.3 ❖ Принцип подачи данных в облачное хранилище. Spark играет роль поточного сервиса

Обработка сложных событий

Обработка сложных событий (англ. *Complex event processing,* или *CEP*) – это еще одна аналитическая система, которая используется для распознавания шаблонов. В 1990-х гг. ее начали использовать в дискретно-событийном моделировании и торговле волатильностью на фондовом рынке. По своей природе она позволяет анализировать живые потоки данных в режиме реального времени. Сотни и тысячи событий, поступающие в систему, фильтруются и очищаются, превращаясь в события высшего уровня – более абстрактные, чем исходные показания датчиков. Системы CEP обеспечивают более быстрый анализ даже по сравнению с поточными процессорами, которые могут обработать событие в пределах нескольких миллисекунд. Но, в отличие от Apache Spark, системы CES обладают меньшей избыточностью и способностью к динамическому масштабированию.

Системы CES используют SQL-подобные запросы, однако заданные шаблоны или правила ищутся не в базе данных, а во входящем потоке. CEP состоит из кортежа (отдельного элемента данных с временной меткой) и использует различные аналитические шаблоны, описанные в начале этой главы, которые хорошо сочетаются со скользящими окнами событий. По своей семантике язык запросов похож на SQL, но работает значительно быстрее обычных баз данных, поэтому все правила и данные хранятся в памяти (обычно внутри многогигабайтной БД). Кроме того, данные должны поставляться современной системой поточных сообщений, такой как Kafka.

Системы CEP поддерживают такие операции, как скользящие окна, соединения и обнаружение последовательностей. Кроме того, как и системы правил, они могут полагаться как на прямое, так и на обратное сцепление. Промышленным стандартом в этой области является система Apache WSO2 CEP, которая в сочетании с Apache Storm способна обрабатывать более 1 миллиона событий в секунду, не используя при этом постоянное хранилище. В системе WSO2 используются SQL-запросы, но она также поддерживает скрипты на языках JavaScript и Scala. Еще одним ее преимуществом является возможность расширения функций с помощью пакета под названием *Siddhi*, который предоставляет следующие сервисы:

- геолокация;
- обработка естественного языка;
- машинное обучение;
- корреляция и регрессия временных рядов;
- математические операции;
- работа со строками и RegEx.

Ниже показан пример запроса к потоку данных на языке SiddhiQL:

```
define stream SensorStream (time int, temperature single);
@name('Filter Query')
fromSensorStream[temperature &gt; 98.6'
select *
insertintoFeverStream;
```

Данные воспринимаются как отдельные события, что позволяет применять сложные правила к миллионам сообщений, поступающих одновременно.

Архитектор должен понимать, в каких ситуациях следует применять системы правил и CEP. Если условие оперирует простыми показаниями, такими как диапазоны температур, система лишена состояния и может быть реализована на основе простого механизма BRMS. Если же система хранит временные данные или ряд состояний, следует использовать CEP.

Lamda-архитектура

Lambda-архитектура пытается найти компромисс между латентностью и пропускной способностью (рис. 11.4). В сущности, она сочетает в себе пакетную и поточную обработку. По аналогии с общей облачной топологией OpenStack и других облачных фреймворков Lambda сохраняет потребляемые данные в репозитории, доступном только для чтения. Такая топология состоит из трех уровней:

- **пакетный уровень** – обычно основан на кластерах Hadoop. Обработка в нем происходит намного медленней, чем на поточном уровне. Принося в жертву латентность, он максимизирует пропускную способность и точность;
- **поточный уровень** – это поток данных, проходящих через память в режиме реального времени. Данные могут теряться, содержать ошибки и приходить не в том порядке. Как мы уже видели, Apache Spark предоставляет отличную систему обработки потоков;
- **сервисный уровень** – здесь происходит сохранение, анализ и визуализация пакетных и поточных результатов. Обычно здесь применяются такие компоненты: Druid (предоставляет средства объединения пакетного и поточного уровней), Apache Cassandra (масштабируемая база данных) и Apache Hive (долговременное хранилище данных).

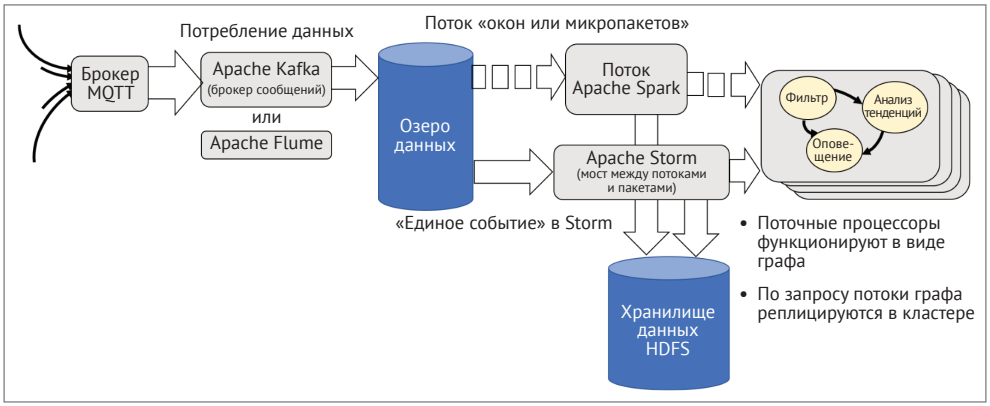


Рис. 11.4 ❖ Сложности Lambda-архитектуры.
Здесь пакетный уровень помещает данные в хранилище HDFS, а поточный уровень передает свои результаты непосредственно системе анализа в режиме реального времени, используя Spark

Lambda-архитектуры по своей природе имеют повышенную сложность, если сравнивать их с другими аналитическими системами. Это гибридный подход, для успешной реализации которого требуются дополнительные ресурсы.

Промышленное применение

Здесь мы постараемся рассмотреть типичные примеры применения вышеописанных систем в отраслях, в которых внедряется интернет вещей и облачный анализ. Чтобы спроектировать подходящее облачное решение и подобрать подходящую аналитическую архитектуру, следует учитывать такие критерии, как масштаб, пропускную способность, необходимость обработки в реальном времени и типы данных.

Это обобщенные примеры. При составлении аналогичной таблицы важно понимать принцип движения данных в системе, а также ее масштаб/производительность (табл. 11.1).

Машинное обучение в интернете вещей

Машинное обучение давно является частью информатики. Математические модели для сбора данных и вычисления их вероятности уходят корнями в начало XIX века, к теореме Байеса и методу наименьших квадратов. В наши дни обе эти методики применяются в моделях машинного обучения, и в этой главе мы кратко по ним пройдемся.

Впервые компьютерные машины и концепция обучения сошлись вместе в начале 1950-х гг., когда Марвин Мински (MIT) представил первые нейросетевые устройства под названием *перцептроны*. Позже в 1969 г. он написал статью, которая была воспринята как критика ограничений нейронных сетей. Конечно, в те годы вычислительные ресурсы были на вес золота. В лучшем случае математики могли рассчитывать на такие ЭВМ как IBMS/360 и CDC. Как вы увидите, именно в 1960-х гг. была разработана большая часть математических моделей для искусственного интеллекта (ИИ) в таких областях, как нейронные сети, метод опорных векторов, нечеткая логика и т. д.

В конце 1960-х и в 1970-х гг., благодаря работе Инго Реченберга под названием «Эволюционная стратегия» (нем. *Evolutionsstrategie*, 1973), внимание исследователей сместилось в сторону эволюционных вычислений, таких как генетические алгоритмы и роевой интеллект. Они возымели определенный успех в решении сложных инженерных задач. Генетические алгоритмы по сегодняшний день используются в машиностроении и автоматическом проектировании программного обеспечения.

Кроме того, в середине 1960-х гг. в качестве разновидности вероятностного ИИ была предложена концепция скрытых марковских моделей, таких как байесовская сеть. Она получила применение в области распознавания жестов и биоинформатике.

Исследования искусственного интеллекта приутихли с урезанием правительственного финансирования, пока в 1980-х гг. не появились логические

системы. Это положило начало новой области ИИ, известной под названием логический ИИ, и породило вспомогательные языки программирования Prolog и LISP, с помощью которых программисты могли с легкостью описывать символьные выражения. Однако исследователи нашли ограничения у этого подхода к искусственному интеллекту: семантика, основанная на логике, противоречит человеческому *мышлению*. Попытки использования *антилогических* и *грязных* моделей для описания объектов тоже не увенчались успехом. Если коротко, то слабо связанные понятия не позволяют точно описать объект. Позже в 1980-х гг. популярность получили экспертные системы. Это разновидность логических систем, которая обучается экспертами в конкретной области с прицелом на четко определенную задачу. Это своего рода BRMS для системы управления. Экспертные системы зарекомендовали себя в корпоративных и бизнес-средах, став первым примером коммерчески успешного продукта на основе ИИ. Вокруг них начали формироваться целые новые отрасли. Этот вид ИИ продолжал развиваться, и компания IBM построила на его основе компьютер *Deep Thought*, который в 1997 г. одолел в шахматном поединке гроссмейстера Гарри Каспарова.

Нечеткая логика впервые упоминается в исследовании Лофти Заде (Беркли, 1965), но лишь в 1985 г. инженеры компании Hitachi продемонстрировали ее успешное применение в системах управления. Это вызвало большой интерес среди японских фирм, производящих автомобили и электронику, которые попытались интегрировать нечеткие системы в реальные продукты. С тех пор эта концепция успешно применяется в системах управления, и позже в этой главе мы обсудим ее во всех деталях.

Экспертные системы и нечеткая логика стали оплотом ИИ, однако разница между их текущими возможностями и тем, что они никогда не смогут делать, становилась все более заметной. В начале 1990-х исследователи пришли к выводу, что экспертные и логические системы в целом неспособны эмулировать разум. В это же время в виде скрытых марковских моделей и байесовских сетей начал развиваться статистический подход к искусственному интеллекту. В сущности, информатика переняла модели, широко используемые в экономике, торговле и исследовании операций.

В 1963 г. Владимир Н. Вапник и Алексей Червоненкис впервые предложили метод опорных векторов (МОВ), хотя популярным он стал уже после периода застоя в ИИ, который длился с 1970-х до начала 1980-х гг. МОВ стал основой для линейной и нелинейной классификации, предоставляя новаторский подход к поиску лучшей гиперплоскости для категоризации наборов данных. Эта методика получила широкое распространение благодаря анализу почерка и вскоре нашла применение в нейронных сетях.

В 1990-х гг. возник интерес к рекуррентным нейронным сетям (англ. *Recurrent neural network,* или *RNN*). Эта уникальная модель отличалась от нейронных сетей глубокого обучения (таких как сверточные нейронные сети) за счет сохранения своего состояния и могла применяться к проблемам, в которых

Таблица 11.1. Типичные примеры применения вышеописанных систем

Индустрия	Применение	Облачные сервисы	Типичная нагрузка	Латентность	Анализ
Производство	• Технологии мониторинга • Обновление старых систем • Отслеживание активов • Автоматизация фабрик	• Панели управления • Массивные хранилища • Озера данных • SDN (гибридная облачная топология) • Низкая латентность	• 500 Гб / в день (на каждый узел фабрики) • 2 Тб / в минуту (на каждую операцию майнинга)	Меньше 1 с	• Рекуррентные нейронные сети • Байесовские сети
Логистика и перевозки	• Геолокационное отслеживание • Отслеживание активов • Мониторинг оборудования	• Панели управления • Журналирование • Хранение	• **Наземный транспорт**: 4 Тб / в день на одну машину (50 датчиков) • **Воздушный транспорт**: 2,5–10 Тб / в день (6000 датчиков) • **Отслеживание активов**: 1 Мб / день (на каждый маяк)	• Меньше 1 с (в реальном времени) • Ежедневно (пакетный режим)	Системы правил
Здравоохранение	• Отслеживание активов • Отслеживание пациентов • Мониторинг здоровья на дому • Беспроводное медицинское оборудование	• Надежность и HIPAA • Частное облако • Хранение и архивация • Балансировка нагрузки	1 Мб / в день (на каждый датчик)	• **Меньше 1 с**: жизненно важные услуги • **Раз в смену**: менее важные услуги	• **Рекуррентные нейронные сети (RNN)** • Деревья принятия решений • Системы правил
Сельское хозяйство	• Отслеживание здоровья и местоположения скота • Химический анализ почвы	• Массивное хранилище и архивация • Передача данных из облака в облако	• 512 Кб / в день на каждое животное • 1000–10000 голов в загоне	• 1 с (в реальном времени) • 10 минут (пакетный режим)	Системы правил

Энергетика	• Умные счетчики • Удаленный мониторинг электроэнергии (солнечные панели, газ, нефть) • Прогноз неисправностей	• Панели управления • Озера данных • Массивные хранилища для прогнозов на основе имеющихся данных; • SDN • Низкая латентность	• 100-200 Гб / в день на каждую ветряную турбину • 1–2 Тб / в день на нефтяную вышку • 100 Мб / в день на умный счетчик	• **Меньше 1 с**: производство электроэнергии • **1 минута**: умный счетчик	• RNN • Байесовские сети • Системы правил
Потребительский рынок	• Диагностика в реальном времени • Обнаружение присутствия • Освещение и отопление/кондиционер • Безопасность • Умный дом	• Панели управления • PaaS • Балансировка нагрузки • Массивное хранилище	• **Видеонаблюдение**: 500 Гб / в день на камеру • **Умные устройства**: 1–1000 Кб / в день на датчик или устройство • **Умный дом**: 100 Мб / в день на дом	• **Видео**: меньше 1 с • **Умный дом**: 1 с	• Сверточные нейронные сети (распознавание изображений) • Системы правил
Торговля	• Мониторинг холодной цепочки • Устройства POS • Системы безопасности • Маяки	• SDN/SDP • Микросегментация • Панели управления	• **Безопасность**: 500 Гб / в день на камеру • **Общее назначение**: 1–1000 Мб / в день на устройство	• **Транзакции с кредитными картами и POS-терминалами**: 100 мс • **Маяки**: 1 с	• Системы правил • Сверточные нейронные сети для безопасности
Умный город	• Умные стоянки • Умный вывоз мусора • Датчики окружающей среды	• Панели управления • Озера данных • Межоблачные услуги	• **Мониторинг электроэнергии**: 2,5 Гб / в день на город (70 000 датчиков • **Парковочные места**: 300 Мб / в день (80 000 датчиков) • **Мониторинг загрязнения**: 350 Мб / в день (200 000 датчиков) • **Мониторинг шума**: 650 Мб / в день (30 000 датчиков)	• **Счетчики электроэнергии**: 1 мин • **Температура**: 15 мин • **Шум**: 1 мин • **Загрязнение**: 10 мин • **Парковочные места**: раз в смену	• Системы правил • Деревья принятия решений

фигурирует понятие времени – например, распознавание звуков и речи. RNN оказали непосредственное влияние на современные модели прогнозирования в интернете вещей, которые мы обсудим позже в этой главе.

Нечто похожее произошло в 2012 г. в области распознавания изображений. Команды, собранные со всего мира, соревновались в распознавании объекта на картинке размером 50×30 пикселей. После маркировки объекта вокруг него следовало начертить прямоугольник. Всего следовало обработать 1 миллион изображений. Победителем стала команда Университета Торонто, которая впервые использовала для решения этой задачи глубокую сверточную нейронную сеть. До этого в области машинного зрения уже применялись другие виды нейронных сетей, однако канадской команде удалось добиться невиданной точности распознавания с частотой ошибок 16,4%. Другая нейронная сеть, разработанная компанией Google, снизила этот показатель до 6,4%. Примерно в то же время Алекс Крижевски создал систему AlexNet, которая значительно ускорила процесс обучения за счет использования графических карт. Все эти модели были построены на основе сверточной нейронной сети и требовали вычислительной мощи, которая была недоступна до появления графических процессоров.

В наши дни искусственный интеллект можно встретить везде: в беспилотных автомобилях, при распознавании речи в Siri, в автоматических системах обслуживания клиентов, медицине и торговле, где модели машинного обучения позволяют отслеживать интерес покупателей по мере их перемещения по магазину (рис. 11.5).

Но какое отношение это имеет к интернету вещей? Дело в том, что IoT-устройства транслируют громадные и непрерывные потоки данных. Множество датчиков передают свои показания, формируя наше представление о системе в целом. Как уже упоминалось ранее, интернет вещей должен стать катализатором к увеличению объема собираемых данных на целый порядок. Иногда информация оказывается структурированной (как, например, в случае с временными рядами), а иногда нет (как в камерах, синтетических датчиках, аудио и аналоговых сигналах). Клиенты хотят принимать рациональные бизнес-решения на основе этих данных: например, оптимизировать операционные и, возможно, капитальные расходы за счет внедрения IoT и машинного обучения (по крайней мере, эту идею им рекламируют). Представьте себе фабрику со множеством взаимосвязанных систем: инструменты сборки, роботы для резьбы по металлу и пластику, еще одна машина для литья под давлением, конвейерные ленты, системы освещения и обогрева, упаковочный автомат, системы управления поставками и инвентаризацией, роботы для перемещения материалов и разные многоуровневые механизмы управления. Все это может быть разбросано по разным помещениям или даже городам. Такие фабрики уже внедрили у себя все традиционные методы повышения эффективности, а их руководство знакомо с трудами У. Эдвардса Деминга; однако следующая индустриальная революция будет связана с интернетом вещей и искусственным интеллектом.

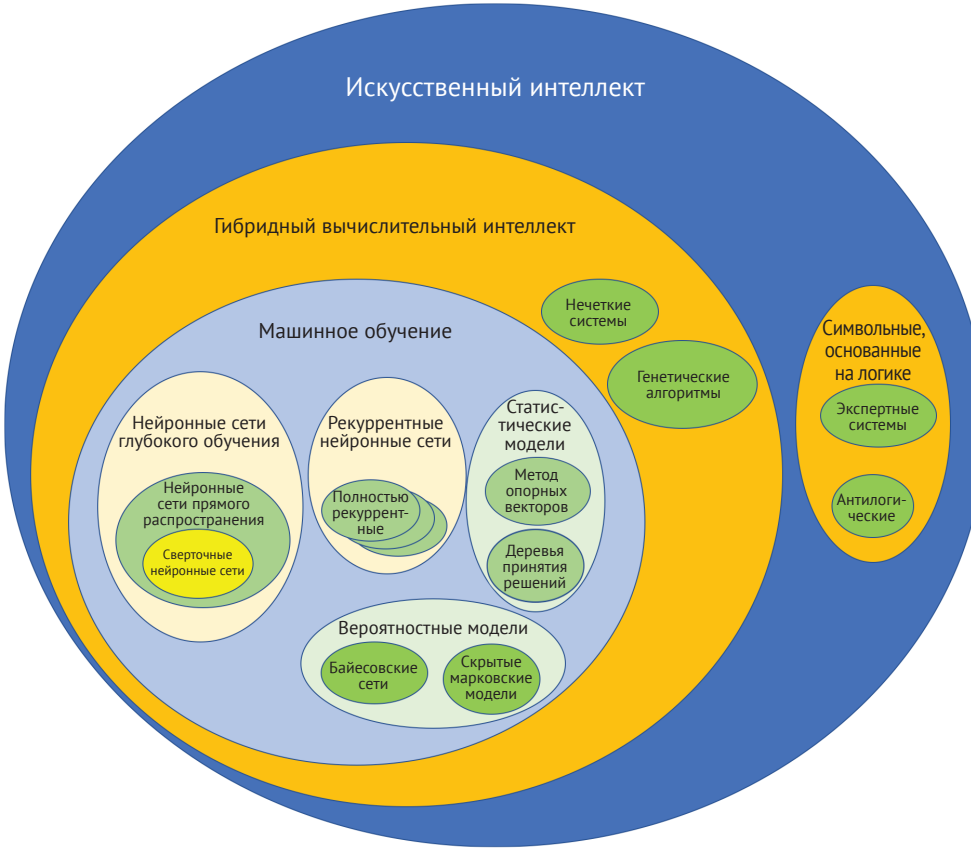


Рис. 11.5 ❖ Спектр алгоритмов искусственного интеллекта

Специалисты знают, что делать при возникновении нештатных ситуаций. Например, оператор сборочного станка с многолетним опытом может определить по его поведению, нуждается ли тот в сервисном обслуживании. Возможно, станок начал скрипеть или уронил несколько деталей за последние дни, что свидетельствует об износе соответствующего механизма. Машинное обучение позволяет обнаруживать и прогнозировать такие элементарные признаки даже раньше, чем это сделал бы человек. Станок можно окружить датчиками и отслеживать как уже произошедшие, так и потенциальные события. Таким образом, мы можем следить за целой фабрикой; принимая миллионы и миллиарды событий от каждого устройства и работника, мы будем *видеть* состояние фабрики в каждый конкретный момент.

При таких объемах данных только решения на основе искусственного интеллекта способны отфильтровать шум и найти полезную информацию. Эта задача не по силам человеку, но с ней вполне могут справиться модели больших данных и машинного обучения.

Модели машинного обучения

Здесь мы рассмотрим конкретные модели машинного обучения, которые получили применение в интернете вещей. Для фильтрации наборов данных нет какого-то единого универсального решения. Каждая модель имеет свои сильные стороны и подходит для использования в определенных сценариях. Целью любого средства машинного обучения является создание прогноза или вывода на основе предоставленной информации. Ваши результаты должны быть более надежными, чем подбрасывание монеты.

Существует два вида систем обучения, на которые стоит обратить внимание:

- **обучение с учителем –** предполагается, что каждая запись в обучающих данных имеет метку. Например, это может быть набор изображений, у каждого из которых есть описание содержимого: кот, собака, банан, автомобиль и т. д. На сегодня многие модели машинного обучения используют этот подход. Обучение с учителем позволяет решать проблемы классификации и регрессии, о которых мы поговорим позже в этой главе;
- **обучение без учителя –** обучающие данные не содержат меток. Очевидно, что этот метод не может определить, что именно изображено на картинке. Он использует математические правила для снижения избыточности. Типичный пример применения – поиск скоплений похожих объектов.

Существует также гибридная модель *с частичным привлечением учителя*, в которой лишь некоторые данные содержат метки. Ее цель – организовать данные и сделать логические выводы.

Машинное обучение имеет три основные области применения:

- классификация;
- регрессия;
- обнаружение аномалий.

Существуют десятки концепций машинного обучения и искусственного интеллекта, которые могут применяться в контексте интернета вещей, большинство из них выходят за рамки этой книги. Мы сосредоточимся на небольшом наборе моделей и попытаемся понять, как они соотносятся между собой, на что они направлены и каковы их сильные стороны. Вы познакомитесь со сценариями использования и ограничениями статистической и вероятностной моделей, глубокого обучения и RNN, так как именно эти области превалируют в сфере ИИ для интернета вещей.

Мы обобщим и подробно изучим такие темы:

- **случайные леса** – статистические модели (быстрые, подходят для систем со множеством атрибутов, необходимых для обнаружения аномалий);
- **байесовские сети** – вероятностные модели;
- **сверточные нейронные сети** – глубокое обучение (модель глубокого обучения для неструктурированных графических данных);
- **RNN** – рекуррентные нейронные сети (модели глубокого обучения для анализа временных рядов).

Некоторые модели больше не применяются в сфере искусственного интеллекта, по крайней мере, в той области IoT, которая нас интересует. В связи с этим мы не станем останавливаться на логических системах, генетических алгоритмах и нечеткой логике.

Для начала следует обсудить базовую терминологию вокруг классификаторов и регрессии.

Классификация

Обсуждение большинства моделей машинного обучения следует начинать с классификации. Классификация – это вид обучения с учителем, в котором данные используются для подбора имен, значений и категорий. Например, нейронная сеть может сканировать фотографии в поисках изображения ботинка. В этой области существует два вида классификации:

- **биномиальная –** если выбирается одна из двух категорий (кофе, чай);
- **множественная –** если существует больше двух вариантов.

Мы воспользуемся Стэндфордским линейным классификатором, чтобы лучше понять концепцию гиперплоскостей (vision.stanford.edu/teaching/cs231n-demos/linear-classify/). На рис. 11.6 показано, как обученная система пытается найти лучшую гиперплоскость для сортировки цветных шариков. Как видите, после нескольких тысяч итераций разделение выглядит почти оптимальным, хотя правая верхняя область остается проблемной, так как соответствующая гиперплоскость включает в себя шарики, принадлежащие верхней гиперплоскости. Это пример недостаточно оптимальной классификации.

Здесь гиперплоскости используются для создания искусственных сегментов. В правой верхней части находится шарик, который должен был входить в одну категорию с двумя верхними шариками, но попал в правый нижний набор.

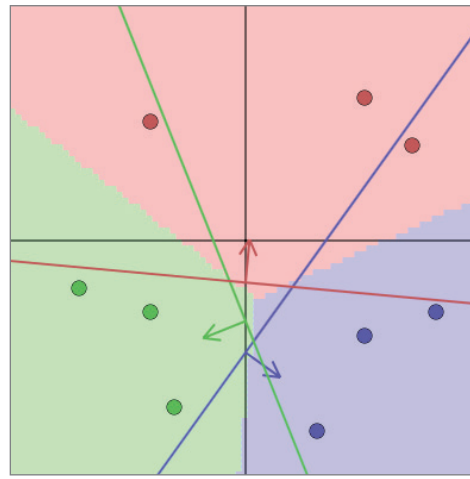


Рис. 11.6 ❖ Недостаточно оптимальная классификация

Заметьте, что в этом примере гиперплоскость представляет собой прямую линию. Эта модель называется *линейным классификатором* и включает в себя такие концепции, как метод опорных векторов (который пытается максимизировать линейность) и логистическая регрессия (которая подходит для биномиальной и множественной классификации). На рис. 11.7 показана линейная биномиальная классификация двух наборов данных: окружностей и прямоугольников.

Линия пытается сформировать гиперплоскость, чтобы разделить переменные на две четкие области. Стоит отметить, что лучшее линейное разделение содержит ошибки.

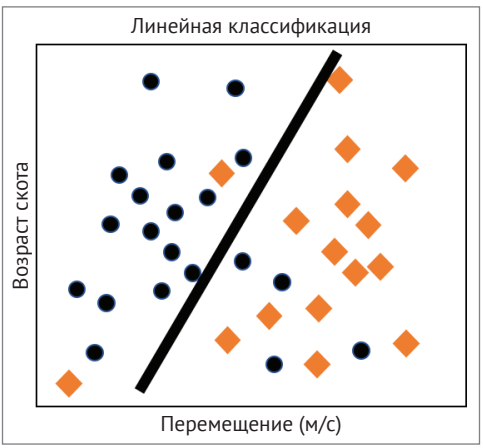


Рис. 11.7 ❖ Линейный классификатор

Нелинейные отношения тоже не являются редкостью в машинном обучении. Они позволяют снизить количество ошибок в линейной модели. На рис. 11.8 показана одновременно линейная и нелинейная классификация. У последней есть одна проблема: она имеет свойство проходить слишком близко к набору точек. Как вы позже увидите, это позволяет добиться повышенной точности с тестовыми наборами, но в реальных условиях этот эффект нивелируется. Ниже представлено сравнение линейного и нелинейного классификаторов.

Регрессия

Классификация занимается прогнозированием конкретного значения (круг или квадрат), тогда как модели регрессии дают прогноз непрерывных величин. Например, регрессионный анализ можно использовать для предсказания средней цены вашего дома, исходя из стоимости всех домов по соседству и в окрестностях.

Для выполнения регрессионного анализа существует несколько методик:

- метод наименьших квадратов;
- линейная регрессия;
- логистическая регрессия.

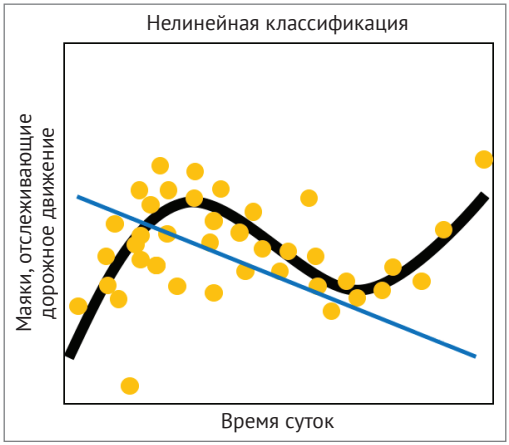


Рис. 11.8 ❖ Здесь полиномиальная кривая n-го порядка пытается построить значительно более точную модель для набора данных. Высокоточные модели имеют свойство хорошо вписываться в тестовые наборы, но при работе с реальными данными их результаты оказывают неудовлетворительными

Случайный лес

Случайный лес (англ. *Random forest*) – это разновидность другой модели машинного обучения под названием *дерево принятия решений*. Как видно на диаграмме в начале этого раздела, дерево принятия решений представляет собой группу обучающихся алгоритмов, которые принадлежат к статистическим методам. Оно принимает несколько переменных и выдает единый результат, который характеризует весь набор. *Набором* называют каждый оцениваемый элемент. Дерево принятия решений оценивает вероятность того, что заданный маршрут основан на предоставленном вводе. Одной из разновидностей этого подхода является алгоритм *CART* (англ. *Classification and Regression Test*), разработанный Лео Брейманом в 1983 г.

Теперь пришло время познакомиться с *бутстрэппингом*. Если вы обучаете лишь одно дерево принятия решений, оно становится восприимчивым к шуму, который в него проникает, и может демонстрировать *предвзятость* (смещение результатов). Если же таких деревьев много, риск предвзятости снижается. Каждое дерево получит случайный набор данных для обучения.

Результатом обучения случайного леса является дерево принятия решений, основанное на произвольном наборе входящих данных и произвольном выборе переменных (рис. 11.9).

Случайные леса расширяют идею бутстрэппинга, выбирая не только произвольный набор данных, но и подмножество критериев, которые учитываются. Это можно наблюдать на рис. 11.9. Такой подход может показаться нелогичным, так как в обучении имеет смысл использовать как можно больше информации. Этому есть два объяснения:

- большинство деревьев являются точными и предоставляют правильные прогнозы для большей части данных;
- потенциальные ошибки возникают в разных участках и разных деревьях.

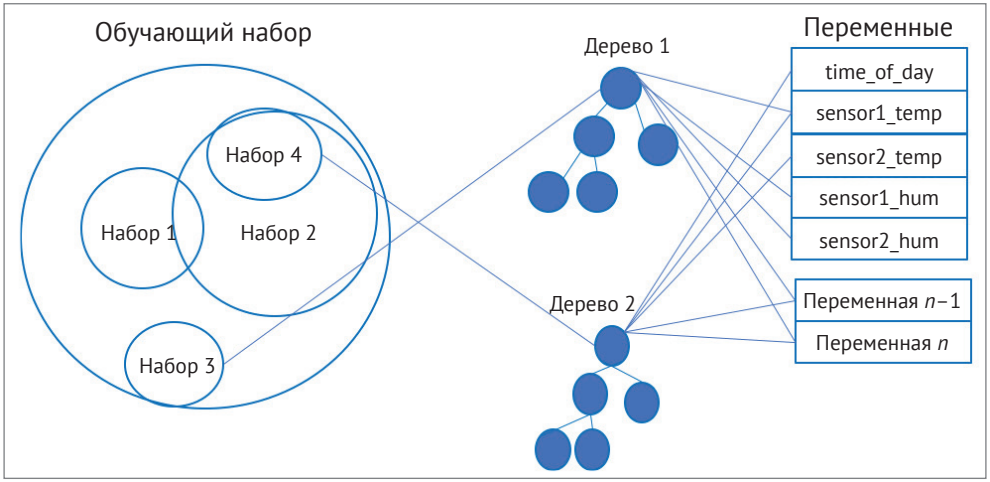


Рис. 11.9 ❖ Модель случайного леса. Здесь формируется два дерева, которые выбирают из набора случайные перемены (но не весь набор целиком)

Это принцип работы *группового мышления* и *решений большинства*. Если результаты нескольких деревьев получены разными путями, но согласуются между собой, и лишь одно дерево не вписывается в эту картину, оно естественным образом примет решение большинства. Эта модель имеет малое расхождение по сравнению с использованием одного дерева принятия решений, которое может оказаться крайне предвзятым. Ниже показан пример с четырьмя деревьями в случайном лесе. Каждое из них обучалось на отдельном наборе данных и выбрало произвольные переменные. В итоге совокупность деревьев возвращает *9*, хотя четвертое дерево дает другой результат.

Вопреки результатам четвертого дерева, большинство, основанное на разных наборах данных, переменных и топологиях, сошлось на том, что логическим ответом должно быть число *9* (рис. 11.10).

Байесовские модели

Байесовские модели являются производными теоремы Байеса 1812 г., которая описывает вероятность возникновения события, исходя из поведения системы в прошлом. Например, какова вероятность отказа устройства, если нам известна его температура?

Теорема Байеса выглядит так:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}.$$

A и *B*– это интересующие нас события. *P*(*A*|*B*) спрашивает, какова вероятность возникновения события *A*, если событие *B* уже произошло? Они не связаны друг с другом и являются взаимоисключающими.

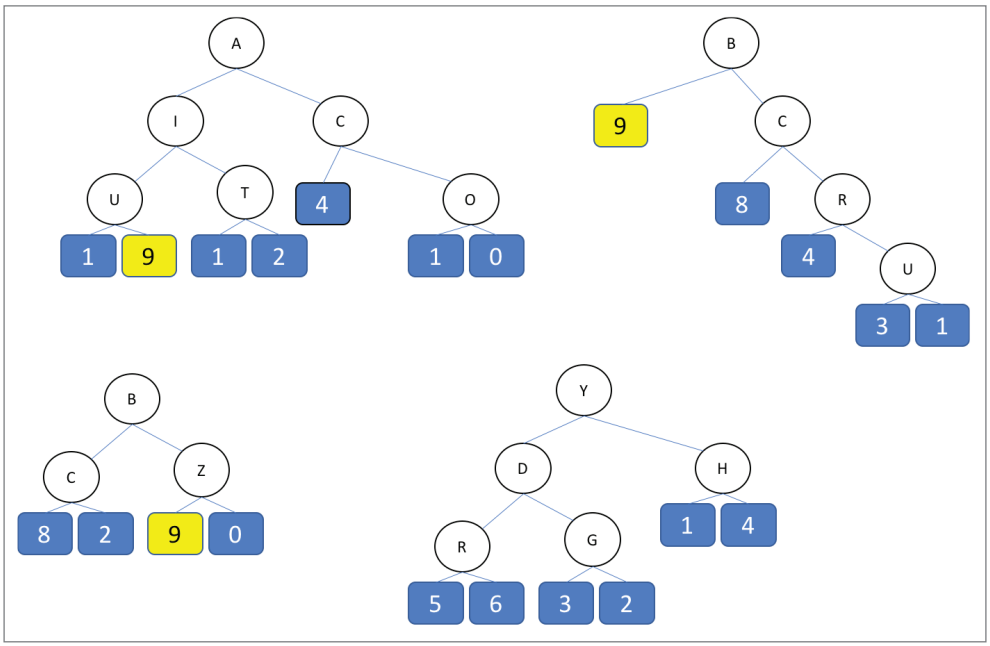


Рис. 11.10 ❖ Решение большинства в случайном лесе. Сразу несколько деревьев, основанных на произвольных наборах переменных, вернули в качестве ответа 9. Выдача похожих результатов при разном вводе обычно ведет к усилению модели

Это уравнение можно переопределить с помощью теоремы полной вероятности, которая заменяет *P*(*B*). Мы можем расширить ее до количества событий *i*. *P*(*B*|*A*) – это вероятность возникновения события *B* при условии, что событие *A* уже произошло. Вот как выглядит формальное определение байесовской теоремы:

$$P(A_i \mid B) = \frac{P(B \mid A_i) \times P(A_i)}{P(B \mid A_1) \times P(A_1) + P(B \mid A_2) \times P(A_2) + \ldots + P(B \mid A_i) \times P(A_i)}.$$

Если мы имеем дело с одной и той же вероятностью и ее дополнением, уравнение можно переписать так:

$$P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B \mid A) \times P(A) + P(B \mid A') \times P(A')}.$$

Давайте рассмотрим пример. Допустим, у нас есть два станка, которые производят идентичные запчасти для устройства. Станок может выйти из строя,

если его температура превысит определенный порог. Для станка *A* вероятность поломки в таких условиях составляет 2%, а для станка *B* – 4%. Станок *A* производит 70% запчастей, а станок *B* – оставшиеся 30%. Если мы возьмем случайную запчасть и она окажется дефектной, какова вероятность того, что она была изготовлена на станке *A* или *B*?

Пусть запчасти называются так же, как и станки, которые их произвели: *A* и *B*. *F* – выбранная дефектная запчасть. Мы знаем, что:

- $P(A) = 0{,}7$;
- $P(B) = 0{,}3$;
- $P(F|A) = 0{,}02$;
- $P(F|B) = 0{,}04$.

Следовательно, вероятность выбора дефектной запчасти, изготовленной на станке A или B, равна:

$$P(A \mid F) = \frac{P(F \mid A) \times P(A)}{P(F \mid A) \times P(A) + P(F \mid B) \times P(B)}.$$

Подставим значения:

$$P(A \mid F) = \frac{0{,}02 \times 0{,}7}{(0{,}02 \times 0{,}7) + (0{,}04 \times 0{,}03)}.$$

Таким образом, $P(A|F) = 53\%$, а дополнение $P(B|F)$ равно $(1 - 0{,}53) = 47\%$.

Байесовская сеть расширяет теорему Байеса, превращая ее в графическую вероятностную модель – а именно, в направленный ациклический граф. Заметьте, что граф направлен в одну сторону, без обратных путей к предыдущим состояниям; это одно из требований байесовской сети (рис. 11.11).

Здесь вероятность каждого состояния оценивается на основе опыта, исторических данных, журнальных записей, тенденций или сочетаний всего вышеперечисленного. Это процесс обучения байесовской сети. Те же правила можно применить к модели обучения в условиях интернета вещей. Модель может прогнозировать поломку станка по мере поступления показаний датчиков, а также делать логические выводы. Например, если датчики сообщают о перегреве, мы можем заключить, что с некоторой вероятностью это может иметь отношение к скорости работы или различным перебоям.

Некоторые разновидности байесовских сетей хорошо подходят для некоторых типов данных и сценариев, но выходят за рамки этого обсуждения:

- наивный байесовский классификатор;
- наивный байесовский-гауссов классификатор;
- байесовские сети доверия.

Байесовская сеть хорошо подходит для сред IoT, которые не могут находиться под полным наблюдением. Она также может пригодиться в условиях с ненадежными данными. Малый размер выборки, наличие шума и потеря показаний оказывают меньшее влияние на байесовские сети, чем на другие виды аналитического прогнозирования. Их недостаток лишь в том, что количество выборок должно быть очень большим. Байесовский метод также хоро-

шо справляется с проблемой переобучения, о которой мы поговорим позже, при рассмотрении нейронных сетей. Стоит также отметить, что байесовские модели хорошо сочетаются с потоковыми данными, столь характерными для интернета вещей. Байесовские сети применяются для обнаружения отклонений в сигналах и потоках данных, связанных по времени; они также помогают находить и отфильтровывать вредоносные сетевые пакеты.

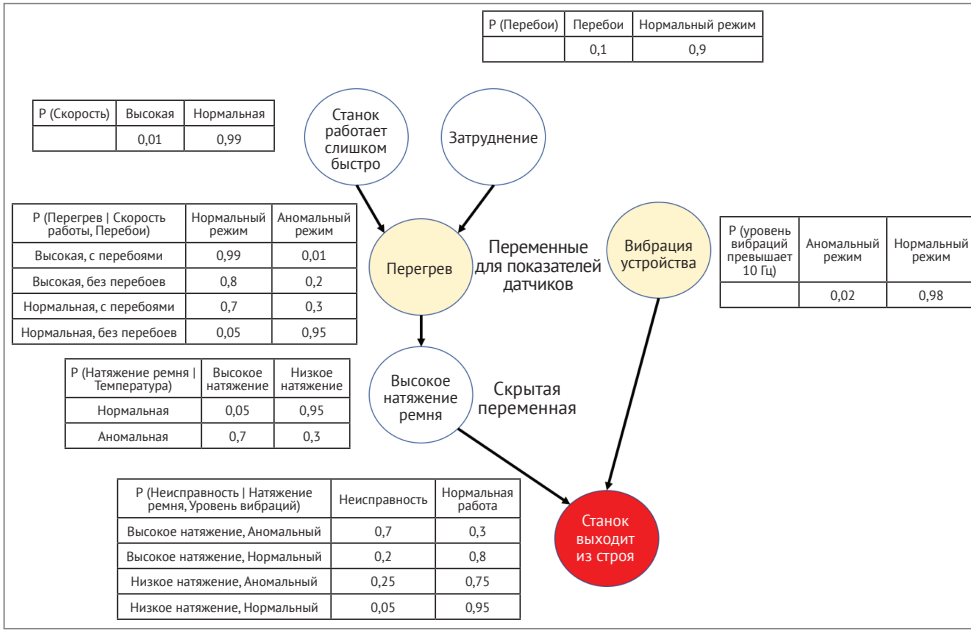


Рис. 11.11 ❖ Модель байесовской сети

Сверточные нейронные сети

Сверточные нейронные сети (англ. *Convolutional Neural Networks,* или *CNN*) – это разновидность искусственных нейронных сетей в машинном обучении. Вначале мы рассмотрим CNN, а затем перейдем к RNN. Сети CNN демонстрируют высокую надежность и точность при классификации изображений; в интернете вещей (особенно в системах безопасности) они используются для визуального распознавания. Это хорошая отправная точка для изучения принципов работы и математического аппарата любой нейронной сети. Любые данные можно представить в виде битовой карты фиксированного размера (например, как изображение размерностью 1024×768 в трех плоскостях). CNN пытается подобрать метку для изображения (например, кот, собака, рыба, птица), используя добавочный набор составных свойств. Простые свойства, характеризующие содержимое изображения, формируются из небольших наборов горизонтальных/вертикальных линий, кривых, теней, векторов градиентов и т. д.

Первый уровень и фильтры

На первом уровне CNN находятся идентификаторы таких свойств: мелкие кривые, мелкие прямые, цветные пятна, мелкие отличительные особенности (в случае с классификацией изображений). Фильтры будут свертываться вокруг изображения в поисках сходства. Сверточный алгоритм берет фильтр и умножает результаты сложений в итоговой матрице. Фильтры активируются, когда свойство генерирует подходящее значение (рис. 11.12).

Рис. 11.12 ❖ Первый уровень CNN.
Для сопоставления по шаблону используются крупные примитивы

Пулинг максимального значения и субдискретизация

Далее обычно идет слой пулинга (или пулинга максимального значения). Он принимает результаты, сгенерированные предыдущим слоем, и возвращает максимальное значение для набора соседних нейронов, которое служит вводом для одного нейрона в следующем слое свертывания. В сущности, это вид субдискретизации. Обычно результатом этого слоя является матрица размером 2×2, состоящая из участков исходной матрицы (рис. 11.13).

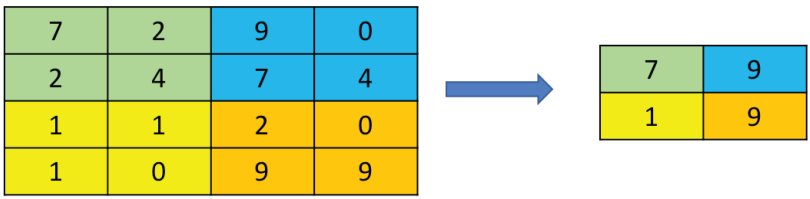


Рис. 11.13 ❖ Пулинг максимального значения.
Пытается найти максимальное значение в диапазоне,
скользящем по изображению

У пулинга есть несколько функций: максимизация (как показано на рис. 11.13), усреднение и другие нетривиальные методы. Цель пулинга максимального значения – доказать, что заданное свойство присуще той или иной части изображения. Нам не нужны конкретные координаты – достаточно лишь общего местоположения. Этот слой заново проходит по плоскостям, с которыми мы имеем дело, что существенно влияет на производительность нейронной сети, а также использование памяти и процессора. Он помогает избежать переобучения. Как показывают исследования, если нейронную сеть настроить специально для работы с изображениями, пренебрегая этим видом субдискретизации, она будет показывать отличные результаты для тех наборов данных, на которых она обучалась, но в реальных условиях от нее не будет никакой пользы.

Скрытые уровни и формальное описание прямого распространения

В качестве ввода второй сверточный слой использует результаты первого слоя, который, как вы помните, принимает исходную битовую карту. На самом деле первый слой возвращает участки двумерного изображения, в которых был обнаружен заданный примитив. Свойства второго слоя представляют собой более сложные структуры, такие как сплайны и кривые. Далее описываются роль нейрона и вычисления, с помощью которых из него извлекается вывод.

Нейрон занимается вводом суммы всех входящих в него весов по отношению к пиксельным значениям. На рис. 11.14 показано, как нейрон принимает ввод из предыдущего слоя в виде весов и значений битовой карты. Сложив веса и значения, он должен пропустить их через функцию активации, чтобы предоставить ввод для следующего слоя.

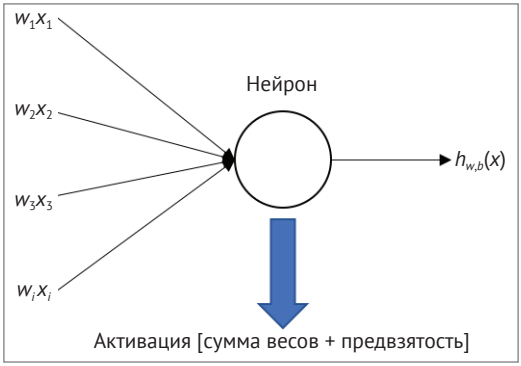


Рис. 11.14 ❖ Базовый элемент CNN.
Нейрон является основной единицей вычисления и принимает на вход веса и значения битовой карты. Срабатывание нейрона зависит от функции активации

Функцию нейрона можно выразить таким уравнением:

$$\sigma\left(\sum_i w_i x_i + b\right).$$

Это можно свести к умножению матриц очень большого размера. Входящее изображение сплющивается в одномерный массив. Предвзятость позволяет влиять на вывод, не затрагивая исходные данные. На рис. 11.15 мы видим пример матрицы весов, умноженной на сплюснутое одномерное изображение с добавлением смещения. Стоит отметить, что в настоящих устройствах на основе CNN этот процесс можно оптимизировать, сместив матрицу весов и добавив единое значение 1,0 к нижней строке вектора битовой карты. В данном примере в итоговой матрице выбирается второе значение – 29,6.

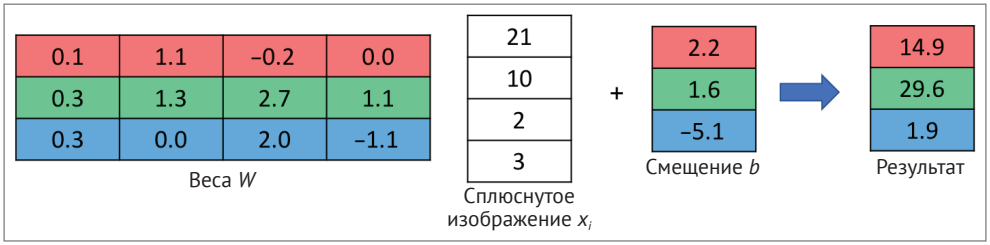


Рис. 11.15 ❖ Соотношение матриц в CNN.
Веса и изображения являются матрицами, которые умножаются и смещаются

Входящие значения умножаются на веса при каждом входе в нейрон. В алгебре это называется простым линейным преобразованием. Сработает нейрон или нет, определяется только после прохождения значения через функцию активации. Цифровые системы на основе транзисторов принимают на вход напряжение; если оно превышает определенный лимит, транзистор включается. Биологическим аналогом является нейрон, поведение которого имеет нелинейную зависимость от ввода. Поскольку мы моделируем нейронную сеть, нам следует выбирать нелинейные функции активации. Среди доступных функций можно выделить следующие:

- логистические (сигмоидные);
- гиперболические;
- *выпрямители* (англ. *Rectified linear unit,* или *ReLU*);
- *экспоненциальные выпрямители* (англ. *Exponential Linear Unit,* или *ELU*);
- синусоидальные.

Сигмоидная функция активации:

$$\sigma(x) = \frac{1}{(1+e^{-x})}.$$

Без сигмоидного слоя (или любой другой функции активации) система свелась бы к линейному преобразованию и имела бы намного меньшую точность при распознавании изображений или шаблонов.

Примеры сверточных нейронных сетей

С примером простых свойств, из которых состоят уровни, можно ознакомиться на сайте TensorFlow (playground.tensorflow.org) (рис. 11.16).

Система, которая там приводится, имеет шесть свойств на первом входящем уровне, за которым следует 33 скрытых уровня с четырьмя нейронами; дальше идут еще две пары нейронов, вторая из которых является завершающей. Все это показано на рис. 11.17. В этой модели свойства пытаются классифицировать точки по их цвету.

Мы попытаемся найти оптимальный набор свойств для описания спирали из двух цветных шариков. Примитивами начального свойства, в сущности, выступают линии и полоски. Они сочетаются между собой и усиливаются путем обучаемого взвешивания, чтобы описать следующий уровень окружностей

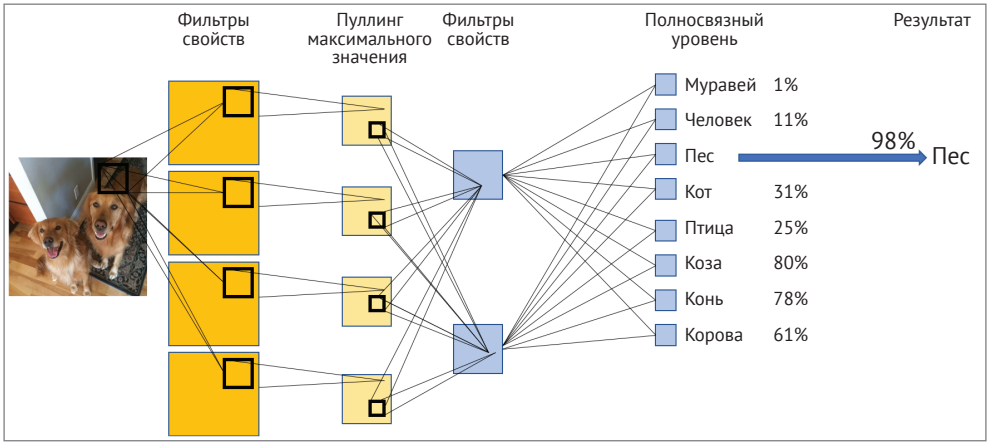


Рис. 11.16 ❖ Четырехуровневая сеть CNN. Изображение свертывается для извлечения общих свойств на основе примитивов, затем используется пуллинг максимального значения, чтобы уменьшить изображение и передать его в качестве ввода для фильтров свойств. В конце маршрута сети находится полносвязный уровень, который выдает лучший вариант

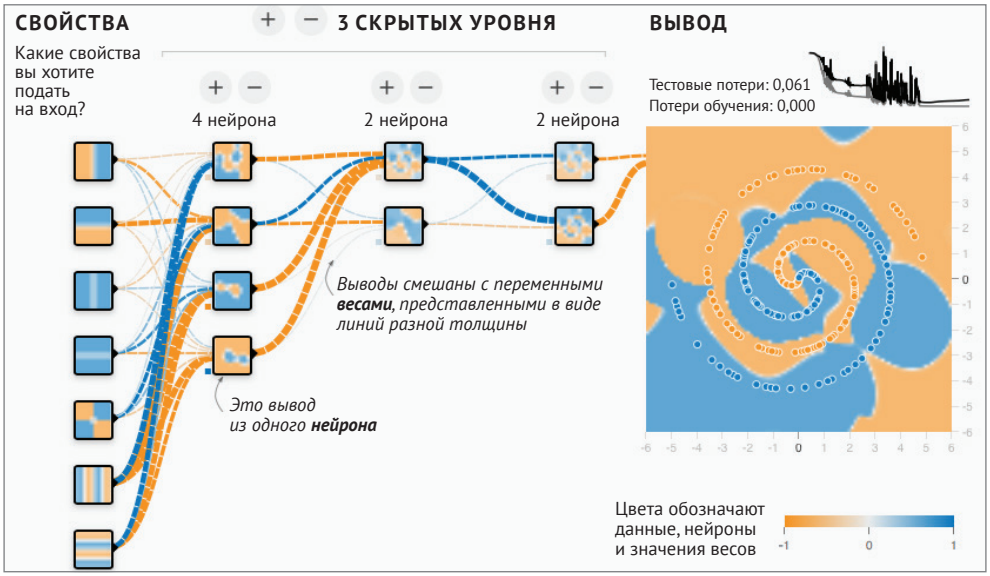


Рис. 11.17 ❖ Пример сети CNN в тестовой среде TensorFlow от Даниэля Смилкова

и пятен. По мере передвижения вправо формируется все более подробное и сложное представление. Этот тест прошел через несколько тысяч эпох, чтобы в итоге показать области, которые описывают спираль, размещенную справа.

В правом верхнем углу можно видеть исходящую кривую, которая представляет частоту ошибок в процессе обучения. Количество ошибок существенно возрастает посреди тестового прогона, так как во время обратного распространения наблюдаются хаотичные и произвольные эффекты. После этого система восстанавливается и оптимизирует итоговый результат. Линии между нейронами символизируют вес в описании спирального шаблона.

Для моделирования сети CNN, представленной выше, используется платформа обучения под названием *TensorFlow Playground*. Здесь обучается четырехуровневая нейронная сеть, цель которой состоит в классификации разноцветных шариков в спирали. Свойства, размещенные слева, являются исходными примитивами: горизонтальное/вертикальное изменение цвета. Скрытые уровни обучаются путем обратного распространения. Степень взвешивания представлена линиями разной толщины, которые ведут к следующему уровню. Результат, полученный после нескольких минут обучения, показан справа.

Последний уровень называется полносвязным, так как каждый его узел должен соединяться с каждым узлом предыдущего уровня. Это делается для того, чтобы, наконец, пометить изображение. Последний уровень анализирует полученные свойства и ввод, устанавливая связи между наборами свойств и конкретными метками. Например, автомобиль будет иметь колеса, стеклянные окна и т. д., тогда как у кота будут глаза, лапы, шерсть и прочие атрибуты.

Обучение и обратное распространение в CNN

Мы уже наблюдали процесс прямой подачи ввода в ходе работы сетей CNN. Обучение в этих сетях основано на обратном распространении ошибок и градиентов, что позволяет снова и снова выводить новые результаты и исправлять искажения. Представленная ниже сеть, включая все уровни пуллинга, функции активации и матрицы, будет использоваться для обратного распространения в попытке оптимизировать или скорректировать значения взвешивания (рис. 11.18).

Здесь функция ошибок вычисляет свой градиент, используя веса нейронной сети. Вычисление градиента подразумевает прохождение в обратном направлении через все скрытые уровни. Ниже показан процесс обратного распространения ошибок (рис. 11.19).

Теперь давайте рассмотрим процесс обучения. Во-первых, в сеть следует передать обучающий набор данных, по которому будет производиться нормализация. Мы еще вернемся к теме обучающих наборов в этой главе, но важно понимать, что это ключевой момент в разработке систем, которые хорошо себя показывают в реальных условиях. Данные будут состоять из изображения и уже известной метки.

Во-вторых, для каждого веса в каждом обучающемся нейроне начальные значения должны быть либо идентичными, либо случайными. Первый прямой прогон возвращает существенное количество ошибок, которые передаются в функцию потерь:

$$W(t) = W(t-1) - \lambda \times \left(\frac{-\partial E}{\partial W}(t) \right).$$

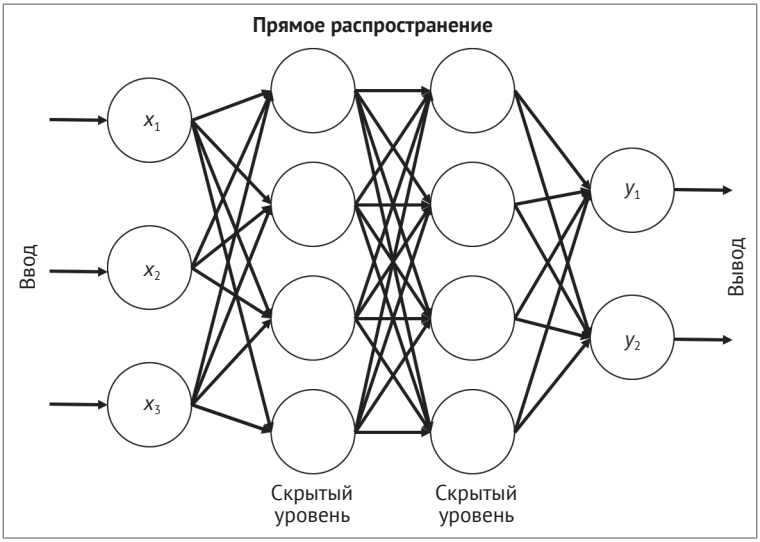


Рис. 11.18 ❖ Прямое распространение в CNN во время обучения и возвращения логического вывода

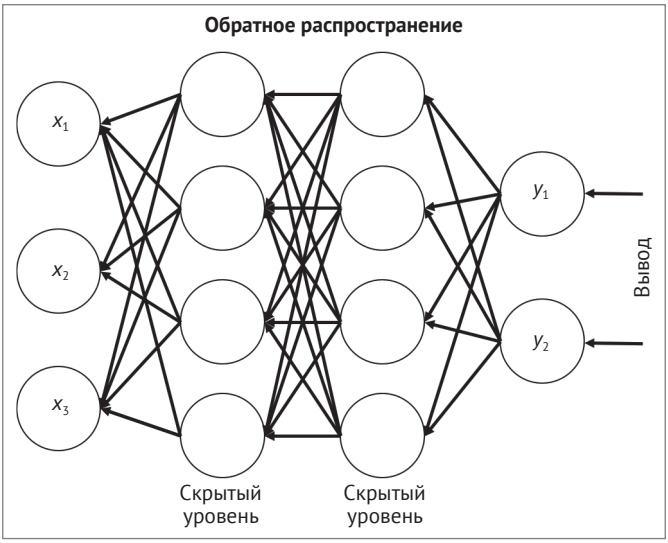


Рис. 11.19 ❖ Обратное распространение в CNN в ходе обучения

Новые веса получаются путем вычитания частичной производной ошибки E по весу W (функция потерь) из предыдущего веса $W(t-1)$. Это также называют *градиентом*. В этом уравнении лямбда обозначает скорость обучения; разработчик сети может ее регулировать. Если скорость высокая (больше 1),

алгоритм будет использовать более крупные отступы в пробном процессе. Благодаря этому сеть может быстрее прийти к оптимальному ответу, хотя с тем же успехом можно получить плохо обученную систему, которая не в состоянии найти решение. Если же значение лямбды небольшое (меньше 0,01), обучение будет состоять из мелких шагов, а схождение произойдет намного позже; при этом может улучшиться точность модели. В следующем примере оптимальная сходимость находится на самом дне кривой, которая описывает ошибки и веса. Если установить слишком высокую скорость обучения, мы никогда не достигнем этой точки и удовлетворимся результатом, который находится *рядом с дном* и направлен в ту или иную сторону (рис. 11.20).

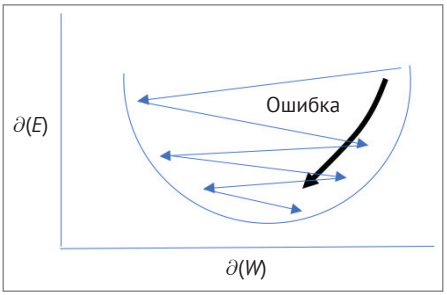


Рис. 11.20 ❖ Глобальный минимум.
Здесь проиллюстрирован базис функции обучения,
которая ищет минимальное значение путем спуска по градиенту.
Точность модели обучения пропорциональна количеству шагов (и времени),
потраченных на схождение к минимуму

Поиск глобального минимума функции ошибки не всегда заканчивается успешно. Иными словами, мы можем «найти» ложный глобальный минимум. Часто алгоритм испытывает проблемы с выходом из локального минимума. На рис. 11.21 показано два минимума один глобальный (настоящий), а другой локальный, который может быть выбран по ошибке.

Количество потерь будет особенно высоким во время начальных прогонов сети. Это можно проиллюстрировать с помощью тестовой среды TensorFlow. Мы снова попытаемся научить нейронную сеть идентифицировать спирали. В начале обучения мы имеем значительные потери – **0,425**. После 1531 эпохи и получения весов для этой сети потери составляют **0,106**.

Как видите, обучение не может устранить все ошибки, поэтому некоторое их количество по-прежнему остается (рис. 11.22).

Здесь мы видим, как по мере продвижения вправо процесс обучения становится все более точным. Слева отчетливо видно сильное влияние свойств с горизонтальными и вертикальными примитивами. После ряда эпох сеть начинает склоняться к верному решению. Но даже после 1531 эпох остаются ошибочные прогоны, которые не дают правильного ответа.

Рис. 11.21 ❖ Ошибки обучения.
Мы видим настоящие глобальные минимум и максимум.
В зависимости от условий, таких как длина шага или даже начальная точка спуска,
CNN может «научиться» выбирать ложный минимум

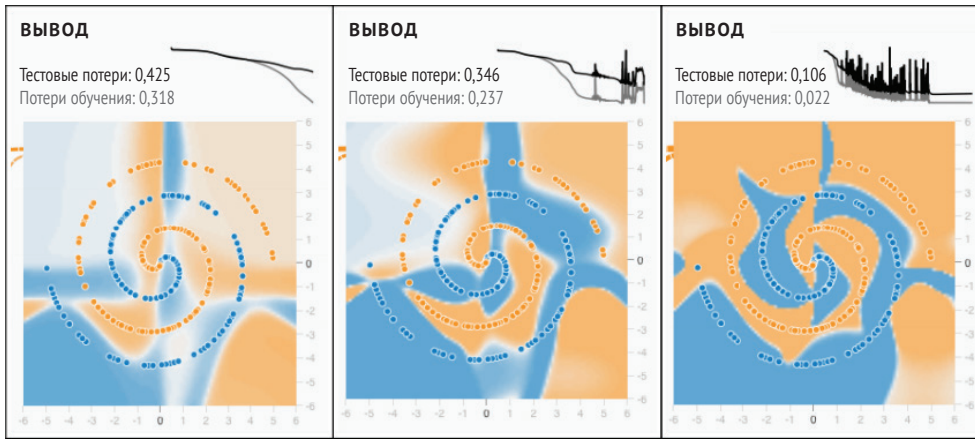


Рис. 11.22 ❖ Пример обучения в TensorFlow от Даниэля Смилкова

Рекуррентные нейронные сети

Рекуррентные нейронные сети (англ. *Recurrent neural networks,* или *RNN*) имеют огромное значение в контексте интернета вещей. Это вид машинного обучения без постороннего вмешательства. В отличие от CNN они не принимают на вход векторы данных фиксированного размера (то есть, двумерные изображения, размер которых заранее известен) и не передают их между уровнями. Рекуррентные сети похожи на сверточные, но обладают одним фундаментальным отличием: они принимают и возвращают векторы. При преобразовании вектора учитывается не только текущий ввод, но и все входящие данные, предоставленные ранее. Таким образом, RNN понимает временную природу

вещей, иными словами, хранит свое состояние. Логический вывод делается на основе текущих и всех предыдущих данных, переданных в сеть.

Сети RNN играют важную роль в интернете вещей, особенно когда речь идет о временных рядах: например, описание сцены на изображении, определение характера наборов текста/переменных или классификация видеопотоков. Данные могут поступать в RNN из массива датчиков, которые предоставляют кортеж вида «время-значение». Такие данные можно послать в рекуррентную сеть. Подобные модели можно использовать в интеллектуальном анализе для поиска неисправностей в промышленных системах автоматизации, поиска аномалий в показаниях датчиков, оценки показаний счетчиков электроэнергии с учетом времени и даже для обнаружения шаблонов в аудиоданных. Еще одним отличным примером являются сигналы, поступающие из промышленных устройств. RNN можно использовать для поиска закономерностей в электрических сигналах и волнах. Сверточная модель столкнулась бы с трудностями в этой ситуации. RNN позволяет предсказать следующий элемент последовательности; если тот не вписывается в спрогнозированный диапазон, это может свидетельствовать о сбое или важном событии (рис. 11.23).

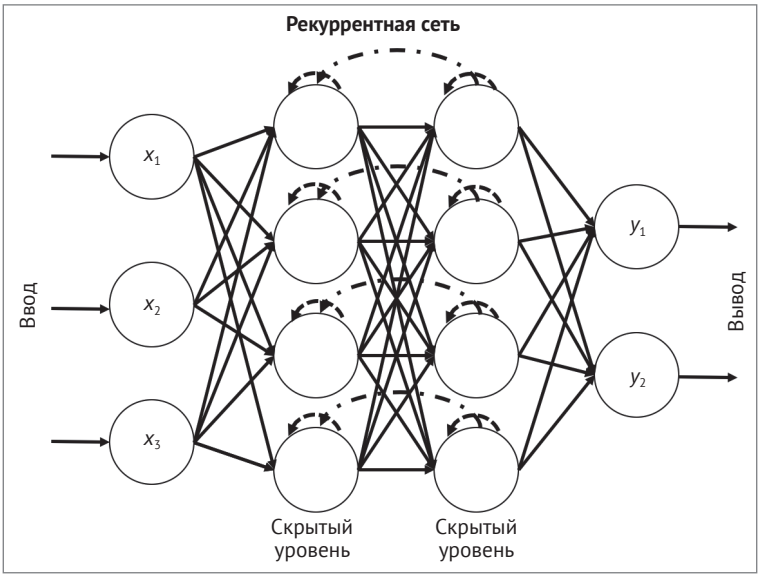


Рис. 11.23 ❖ Основное отличие RNN от CNN заключается в учете времени или порядка элементов в последовательности

Нейрон в рекуррентной сети выглядит так, будто он замкнут сам на себя. По своей сути RNN является набором состояний, которые уходят в прошлое. Это становится очевидно, если представить RNN на уровне отдельных нейронов (рис. 11.24).

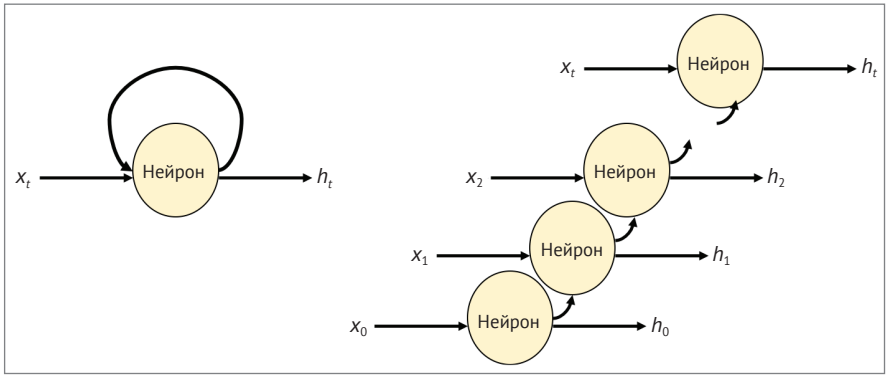


Рис. 11.24 ❖ Нейроны в RNN. Здесь показано, как ввод предыдущего шага, x_{n-1}, передается следующему шагу, x_n. Это базис рекуррентного алгоритма

Проблема сетей RNN состоит в том, что, по сравнению с CNN или другими моделями, они не так легко поддаются обучению. Как вы помните, в CNN для обучения и усиления моделей применяется обратное распространение. В RNN нет такого понятия. Любые данные, попадающие в RNN, содержат уникальную временную метку. Это приводит к проблеме исчезновения градиента, которую мы упоминали ранее, и снижает скорость обучения сети, делая ее бесполезной. Этой проблеме подвержены и сверточные модели, однако RNN, ввиду своей глубины, может уходить назад на множество итераций, тогда как CNN обычно имеет лишь несколько скрытых уровней. Например, при анализе структуры предложения, такого как «*быстрая коричневая лиса перепрыгнула через ленивую собаку*», RNN растянется на семь уровней. Проблему исчезновения градиента можно представить следующим образом: если веса в сети небольшие, градиент будет сжиматься экспоненциально, пока не исчезнет совсем. Если взять более крупные компоненты весов, градиент экспоненциально расширится и, возможно, разорвется (не поместится в память). Разрыв неминуемо приведет к сбою, хотя обычно, пока этого не произошло, градиент урезают или ограничивают. Компьютеру сложнее справиться с этой проблемой.

Одним из способов преодоления этого эффекта является использование функции активации ReLU, которую мы упоминали в разделе о CNN. Ее результатом может быть либо 0, либо 1, поэтому она не склонна к исчезновению градиентов. Еще один вариант связан с концепцией *долгой краткосрочной памяти* (англ. *Long Short-Term Memory,* или *LSTM*), которая была предложена исследователями Зеппом Хохрайтером и Юргеном Шмидхубером в журнале Neural Computation, 9(8):1735-1780, 1997. LSTM решает проблему исчезновения градиента и позволяет обучать RNN. Нейрон в рекуррентной сети состоит из трех или четырех *вентилей*, благодаря которым он может хранить информацию о состоянии. Эти вентили управляются логистической функцией со значениями в диапазоне от `0` до `1`:

- **вентиль забывания K** – определяет меру сохранения значения в памяти;
- **входной вентиль W** – определяет, насколько новое значение повлияет на память;
- **выходной вентиль R** – определяет меру того, в какой степени значение, находящееся в памяти, используется при расчете результата функции активации.

Можно заметить, что эти вентили по своей природе являются аналоговыми, так как они оперируют мерами, а не дискретными значениями. Ячейка LSTM запирает ошибки в памяти блока. Это называется *каруселью ошибок* и позволяет ячейке LSTM распространять ошибки в обратном направлении на протяжении долгого времени. Это напоминает следующую логическую структуру, в которой нейрон внешне функционирует так же, как в CNN, но внутри он имеет память и хранит свое состояние. Ячейка LSTM в рекуррентной сети проиллюстрирована на рис. 11.25.

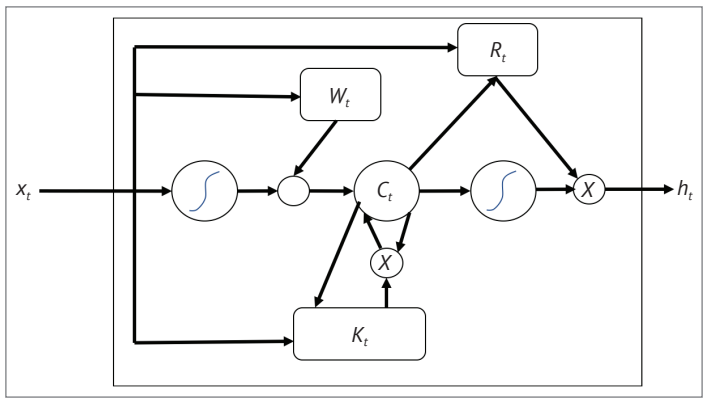


Рис. 11.25 ❖ Ячейка LSTM.
Здесь базовый алгоритм RNN использует внутреннюю память для обработки произвольных входящих последовательностей

В процессе обучения RNN накапливает память. На диаграмме это показано в виде уровня состояния, сразу под скрытым уровнем. RNN не ищет одни и те же шаблоны в изображении или битовой карте, как это делает CNN; вместо этого поиск состоит из нескольких последовательных шагов (которые можно привязать к времени). Скрытый уровень и дополняющий его уровень состояния показаны на рис. 11.26.

Вы можете заметить огромный объем вычислений при обучении с использованием логистических методов LSTM. Кроме того, обратное распространение выглядит менее легковесным, чем в CNN. Процесс обучения подразумевает обратное распространение градиентов по сети, вплоть до начального состояния. Хотя, чем дальше градиент уходит в прошлое, тем меньшее влияние он оказывает на результат.

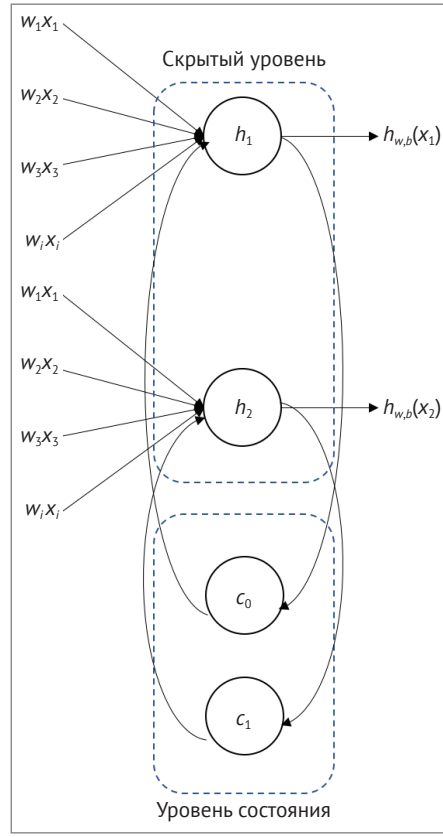


Рис. 11.26 ❖ Скрытые уровни получают данные из предыдущих шагов и служат дополнительным вводом для следующего шага

Проиллюстрировать работу RNN можно на примере анализа сигналов. В промышленных условиях часто собираются и накапливаются данные о сигналах, на основе которых затем пытаются обнаружить дефектные устройства или повышение температуры в некоторых компонентах. Показания датчиков проходят через дискретизацию или анализируются методом Фурье. Затем можно проверить частотные компоненты на предмет характерных отклонений. На рис. 11.27 показана простая синусоида, которая описывает штатное поведение устройства с литым валом или подшипниками. Мы также видим появление двух отклонений (аномалия). Для поиска отклонений в сигналах с помощью гармоник используют *быстрое преобразование Фурье* (англ. *fast Fourier transform,* или *FFT*). Здесь в качестве дефекта выступает высокочастотный всплеск, похожий на дираковскую дельту или импульсную функцию.

Рис. 11.27 ❖ Пример использования RNN. Здесь в качестве ввода можно было бы использовать форму волны с отклонением из области звукового анализа

Как видите, FFT замечает лишь несущую частоту, игнорируя отклонения (рис. 11.28).

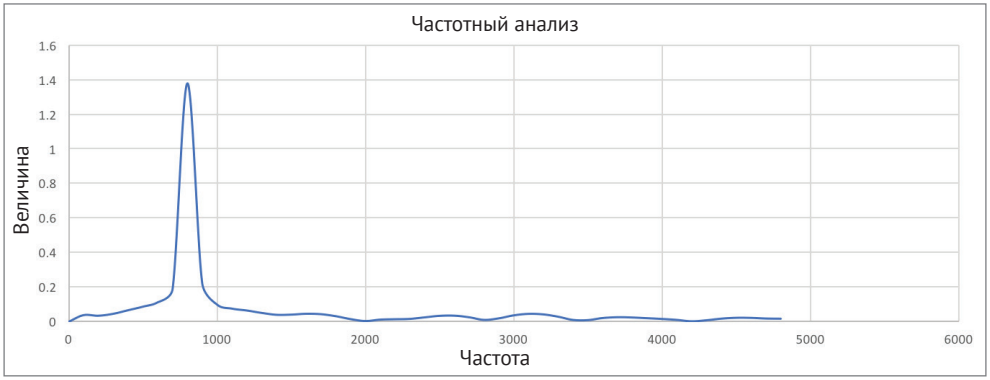


Рис. 11.28 ❖ Огромный частотный всплеск в результате преобразования FFT

Обучение RNN для обнаружения определенного тона или аудио-частоты с привязкой по времени – это довольно простой пример. В данном случае RNN может заменить собой FFT, особенно если для классификации системы используются множественные последовательности частот или состояний; такой метод идеально подходит для распознавания звуков или речи.

Промышленные средства диагностического обслуживания используют этот вид анализа сигналов для поиска неисправностей, связанных с температурой или вибрацией. Но, как мы видим, этот традиционный подход имеет свои ограничения. Модели машинного обучения (особенно RNN) могут применяться для выявления определенных (частотных) компонентов во входящем потоке данных, что позволяет находить неисправные участки, как это показано на рис. 11.28. Однако сырые данные редко когда бывают настолько чистыми, как синусоида, которую мы взяли за пример. Обычно информация содержит довольно много шума с периодическими потерями.

Еще один пример применения RNN связан с объединением датчиков в медицине. Медицинские приборы, такие как глюкометры, пульсометры, датчики падения, респирометры и инфузионные насосы, передают периодические или поточные данные. Все эти датчики являются независимыми друг от друга, но вместе они формируют общую картину здоровья пациента. Они также привязаны ко времени. RNN может агрегировать эти неструктурированные данные и спрогнозировать здоровье человека на основе его дневной активности. Это может пригодиться для отслеживания самочувствия дома, в спортивной подготовке, реабилитации и гериатрическом уходе.

При работе с рекуррентными сетями следует проявлять осторожность. Они могут выдать хорошие результаты на основе временных рядов и спрогнозировать колебания и волновые эффекты, однако они с трудом поддаются обучению, а их поведение может быть хаотичным.

Обучение и получение логических выводов в интернете вещей

Нейронные сети придают компьютерам некую «человечность» в области восприятия, позволяют им распознавать шаблоны и выполнять классификацию. Однако только в процессе обучения можно получить модель, которая допускает лишь незначительные потери, имеет адекватную производительность и не является «переобученной». В мире интернета вещей большой проблемой является латентность, особенно когда речь идет об инфраструктуре, играющей важную роль в безопасности. Еще один фактор – ограниченные ресурсы. Большинство пограничных вычислительных устройств, существующих на сегодняшний день, не могут использовать технологии аппаратного ускорения, такие как *GPGPU* (англ. *General-Purpose Computationon Graphics Hardware – неспециализированные вычисления на графических процессорах*) и *FPGA* (англ. *Field-programmable gate array – программируемая пользователем вентильная матрица*), для выполнения сложных операций линейной алгебры и вычислений с плавающей запятой, которыми пронизаны нейронные сети. Данные можно передать в облако, но это чревато большими задержками и расходами на трафик. Консорциум OpenFog разрабатывает фреймворк, который позволяет выделять пограничным туманным узлам дополнительные вычислительные ресурсы и использовать их на время выполнения громоздких алгоритмов.

Пока что обучение остается прерогативой облачных платформ, которые располагают необходимыми вычислительными ресурсами и способны создавать тестовые наборы. Пограничное устройство должно сообщать своему облаку о неудачной работе обученной модели или поступлении новых данных, которые требуют дополнительного обучения. Облако позволяет многократно развертывать одну и ту же обученную модель, что является плюсом. Возможно также обучение на региональной основе с предвзятостью; такой подход мотивируется тем, что туманные узлы в некоторых регионах могут быть более чувствительными к определенным шаблонам, зависящим от окружения. Например, показания температуры и влажности на производстве могут трактоваться по-разному в арктическом и тропическом климате.

Пограничные устройства больше подходят для выполнения уже обученных моделей в режиме вывода. Однако развертывание систем, генерирующих логические выводы, требует тщательного планирования. Некоторые сверточные сети, такие как AlexNet, принимают 61 миллион параметров, потребляют 249 Мб памяти и выполняют 1,5 миллиарда операций с плавающей запятой для классификации одного единственного изображения.

Понижение точности, сокращение и другие методики, применяющиеся для выполнения начального эвристического анализа графических данных, лучше подходят для пограничных устройств. Также может помочь предварительная подготовка данных для дальнейшего анализа. Например:

- **фильтрация** – анализу подлежат только данные, соответствующие определенным критериям (время, актуальные события);
- **очистка** – сокращение, обрезание, отсечение данных до актуального содержимого;
- **сегментация** – сведение изображения к оттенкам серого, чтобы подготовить его для CNN и сэкономить трафик.

Анализ данных в IoT и сравнение/оценка методов машинного обучения

Алгоритмы машинного обучения получили свое применение в интернете вещей. Типичным примером является ситуация, когда необходимо получить какой-нибудь осмысленный вывод из данных, поступающих из множества потоков. Если ваша система чувствительна к задержкам и количество датчиков не слишком велико, вам может хватить простой системы правил, установленной на границе сети. В других случаях, если требования к латентности не столь строгие, потоки данных можно направлять в облачный сервис и применять правила уже там. Если же вы имеете дело с огромными объемами (неструктурированных) данных, которые следует анализировать в режиме реального времени, для решения наиболее сложных проблем стоит подумать об использовании машинного обучения.

В этом разделе подробно рассмотрим некоторые рекомендации относительно развертывания аналитических систем на основе машинного обучения и сценарии, в которых применение таких систем может быть оправданным (табл. 11.2).

Таблица 11.2. Анализ данных в IoT и сравнение/оценка методов машинного обучения

Модель	Лучшее применение	Худшее применение и побочные эффекты	Требования к ресурсам	Обучение
Случайные леса (статистические модели)	• Обнаружение аномалий • Системы с тысячами точек выбора и сотнями входов • Регрессия и классификация • Обрабатывает смешанные типы данных • Игнорирует потерянные данные • Линейно масштабируется вместе с вводом	• Извлечение свойств • Анализ с учетом времени и порядка следования	Низкие	• Обучение на основе методов агрегации для максимальной эффективности • Обучение без предвзятости с небольшим количеством ресурсов • В основном под надзором
RNN (нейронные сети, основанные на временных рядах и последовательностях)	• Прогнозирование событий на основе последовательности • Шаблоны в поточных данных • Временные ряды • Хранит предыдущие состояния для прогнозирования последующих (электрические сигналы, аудио, распознавание речи) • Неструктурированные данные • Входящие переменные могут зависеть друг от друга	• Анализ изображений и видео • Системы, требующие применения тысяч свойств	• Очень высокие при обучении • Высокие при вычислении логических выводов	• Обучение может быть более громоздким, чем в CNN • Очень сложные в обучении • Обучение с учителем
CNN (глубокое обучение)	• Прогнозирование объекта на основе окружающих значений • Распознавание шаблонов и свойств • Распознавание двумерных изображений • Неструктурированные данные • Входящие переменные могут зависеть друг от друга	• Прогнозирование на основе времени и порядка следования • Системы, требующие применения тысяч свойств	• Очень высокие при обучении (точность вычислений с плавающей точкой, большие обучающие наборы, серьезные требования к памяти) • Высокие при вычислении логических выводов	С учителем и без
Байесовские сети (вероятностные модели)	• Неполные наборы данных, возможно, с шумом • Шаблоны в поточных данных • Временные ряды • Структурированные данные • Анализ сигналов • Быстрое создание моделей	• Предполагается, что все входящие переменные являются независимыми • Плохо работает с многоуровневыми данными	Низкие	• Требуется небольшое количество обучающих данных по сравнению с другими нейронными сетями

Этап обучения:

- для случайных лесов используйте методики агрегации, чтобы создавать композиции;
- при использовании случайных лесов максимально увеличивайте количество деревьев принятия решений;
- избегайте переобучения, так как оно может искажать ваши модели в реальных условиях. Модель можно усилить с помощью таких методик, как регуляризация или даже внедрение шума;
- не выполняйте обучение на пограничных устройствах;
- исчезновение градиента приводит к ошибкам. По своей природе этой проблеме подвержены сети RNN.

Работа с реальными данными:

- обновляйте модель с использованием новых наборов данных по мере их поступления. Поддерживайте обучающий набор в актуальном состоянии;
- модели, выполняющиеся на пограничных устройствах, можно усилить, используя более комплексные и крупные наборы данных в облаке;
- работу нейронных сетей в облаке и на пограничных устройствах можно оптимизировать с минимальными потерями, используя такие методики, как урезание узлов и понижение точности.

Заключение

Эта глава служит кратким введением в анализ данных для интернета вещей в облачных и туманных платформах. Это тот этап, на котором данные извлекаются из потоков, генерируемых миллионами и миллиардами датчиков. Для поиска скрытых закономерностей и прогнозирования на основе громадных объемов информации применяются научные методы. Чтобы быть полезным в принятии жизненно важных решений, анализ должен выполняться в режиме реального времени или близком к нему. Вы должны понимать, какая проблема перед вами стоит и какие данные необходимы для ее решения. Только в этом случае вам удастся спроектировать хороший аналитический процесс. В этой главе было рассмотрено несколько моделей анализа данных, а также четыре актуальных области машинного обучения. Эти механизмы являются сердцем IoT-систем, позволяя им извлекать какой-то смысл из гигантских потоков информации по мере их поступления. Модели машинного обучения способны прогнозировать будущие события, основываясь на уже известных шаблонах. Вы могли видеть, как все эти возможности реализовывались в хорошо обученных сетях RNN и CNN. Как архитектор вы должны учитывать все вместе: процессы обработки и сохранения данных, модели и обучение.

В следующей главе мы комплексно подойдем к вопросу безопасности в интернете вещей. Будут рассмотрены все уровни: от датчика до облака. Мы рассмотрим реальные примеры атак на IoT, произошедшие за последние годы, а также методы, которые позволят вам противостоять им в будущем.

Глава 12

Безопасность интернета вещей

Первая глава этой книги раскрыла масштаб, темпы роста и потенциал **интернета вещей** (**IoT**). На сегодняшний день существуют миллиарды устройств, соединяющих аналоговый мир с интернетом, и за короткое время это число увеличилось на десятки процентов, делая IoT самым крупным объектом для атаки на планете. Уже сейчас происходит разработка, развертывание и глобальное распространение эксплойтов и вредоносов, которые усложняют жизнь бесчисленным предприятиям, сетям и отдельным людям. Мы как архитекторы должны разбираться стеке технологий IoT и уметь их защищать. Мы в ответе за безопасность интернет-устройств, которые никогда раньше не подключались к сети.

Отдельную проблему представляют экземпляры IoT, о безопасности которых думали в последнюю очередь. Часто такие простые системы, как датчики, настолько ограничены, что реализация механизмов защиты промышленного уровня, традиционных для ПК, оказывается слишком сложной, а иногда и вовсе невозможной. Мы обращаемся к теме безопасности уже после знакомства с остальными технологиями, но она так или иначе уже затрагивалась в каждой главе этой книги.

В этой главе мы рассмотрим особо злостные виды атак на интернет вещей и покажем, насколько слабо защищены IoT-устройства и сколько вреда еще можно причинить. Далее мы поговорим об организации безопасности на каждом уровне стека: аппаратном, коммуникационном и сетевом. Мы также уделим внимание программно-определяемым периметрам и блокчейнам, которые используются для защиты ценных данных в IoT. В конце этой главы рассматривается законопроект об улучшении кибербезопасности, принятый в США в 2017 г. (Cybersecurity Improvement Act), и его потенциальные последствия для IoT-устройств.

Самое важное в безопасности – применять ее на всех уровнях: от датчиков до коммуникационных систем, маршрутизаторов и облачных платформ.

Общеупотребительные понятия кибербезопасности

В мире кибербезопасности есть устоявшийся набор определений, которые описывают разные типы атак и меры предосторожности. В этом разделе мы кратко пройдемся по жаргону и терминам, принятым в этой индустрии, и дальше будем использовать их на протяжении всей главы.

Термины, связанные с атакой

Ниже перечислены термины и определения, относящиеся к разным атакам и киберугрозам:

- **атака за счет усиления** – умножает количество трафика, направленного жертве. Для проецирования атаки злоумышленники часто используют легальные сервисы, такие как NTP, Steam или DNS. Коэффициент умножения в NTP может достигать 556, а DNS может увеличить объем трафика в 179 раз;
- **подмена ARP** – выражается в отправке поддельного ARP-сообщения, что позволяет злоумышленнику связать свой MAC-адрес с IP безвредной системы;
- **сканирование баннеров** – эта методика обычно используется для составления списка систем в сети, с помощью которых можно собрать информацию о потенциальных целях атаки. Для этого анализируются данные об ОС и компьютере, возвращенные в ответ на HTTP-запросы (например, `nc www.target.com 80`);
- **ботнеты** – взломанные и зараженные устройства, подключенные к интернету, контролируются для выполнения совместных задач – обычно для отправки запросов в унисон, чтобы сгенерировать громадный трафик с разных клиентов. Возможна также рассылка спама и шпионского ПО;
- **простой перебор** – получение доступа к системе или взлом шифрования методом проб и ошибок;
- **переполнение буфера** – использует ошибку или дефект в рабочем программном обеспечении, направляя в буфер или блок памяти данные, объем которых превышает выделенное пространство. Это позволяет перезаписать другую информацию, хранящуюся в соседних адресах. Злоумышленник может поместить туда вредоносный код и выполнить его, перенаправив на него указатель текущей инструкции. Компилируемые языки, такие как C и C++, особенно подвержены переполнению буфера, так как у них нет внутренней защиты. Большинство подобных ошибок возникают из-за плохо написанного кода, который не проверяет границы вводимых значений;
- **C2** – управляющий сервер, который рассылает команды ботнетам;
- **корреляционный анализ энергопотребления** – четырехэтапная атака, которая позволяет обнаружить секретные криптографические клю-

чи, хранящиеся на устройстве. Сначала анализируется динамическое энергопотребление; показатели записываются для каждой фазы обычного процесса шифрования. Затем целевому устройству передают для шифрования несколько простых текстовых фрагментов и записывают потребление энергии. Дальше взламываются небольшие отрезки ключа (подключи) путем перебора всех возможных комбинаций и вычисления коэффициента корреляции Пирсона между смоделированным и реальным током. В конце собирается самый удачный подключ, который используется для получения целого ключа;

- **перебор по словарю** – взлом сети путем систематического ввода имен пользователей и паролей из готового словаря;
- **DDoS-атака** – попытка нарушить работу интернет-сервиса или сделать его недоступным путем отправки ему большого количества запросов из разных (распределенных) источников;
- **внесение неисправностей** – эта атака заключается в отправке устройству дефектных или нестандартных данных и наблюдении за его реакцией. Например, если устройство начинает плохо работать или показывает признаки неисправности, атака могла выявить слабое место;
- **атака посредника** – распространенная разновидность атак, которая состоит в размещении устройства посреди коммуникационного потока между двумя ничего не подозревающими сторонами. Устройство отслеживает, фильтрует и вычленяет передаваемую информацию, отсылая принимающей стороне ее видоизмененную копию. Злоумышленник может действовать изнутри, выполняя роль ретранслятора, или просто прослушивать данные снаружи, оставляя их без изменения;
- **NOP-сдвиг** – последовательность ассемблерных инструкций NOP, которая позволяет «сдвинуть» указатель текущей инструкции процессора в область вредоносного кода. Обычно является частью переполнения буфера;
- **атака повторного воспроизведения** – сетевая атака со злонамеренным повторением или циклическим воспроизведением данных со стороны оригинального отправителя или злоумышленника, который эти данные перехватывает, сохраняет и передает по своему усмотрению;
- **эксплойт для удаленного выполнения кода** (англ. *remote code execution,* или RCE) – позволяет злоумышленнику выполнять произвольный код. Обычно происходит в виде переполнения буфера по HTTP или другому сетевому протоколу с внедрением вредоносного кода;
- **возвратно-ориентированное программирование** (ROP-атака) – это сложный эксплойт, позволяющий разрушить систему безопасности с помощью неисполняемой памяти или путем выполнения кода из памяти, предназначенной только для чтения. Если злоумышленник получит доступ к стеку процессора через переполнение буфера или каким-то другим путем, он сможет переходить к оригинальным, уже существующим

цепочкам инструкций. Злоумышленника интересуют инструкции для выполнения запросов к устройствам, которые можно использовать для подготовки совместных атак;

- **атака возврата в библиотеку** – эта атака начинается с переполнения буфера. Злоумышленник вставляет в память процесса инструкции перехода к libc или другой популярной библиотеке с целью прямого выполнения системных вызовов. Обходит защиту, основанную на неисполняемой памяти и безопасных адресных пространствах. Это разновидность ROP-атаки;
- **руткит** – как правило, вредоносное программное обеспечение (хотя часто применяется для разблокирования смартфонов), позволяющее скрыть присутствие других программ. В руткитах используется несколько узконаправленных методик, таких как переполнение буфера в компонентах ядра, гипервизоре и пользовательских приложениях;
- **атака по сторонним каналам** – извлечение информации из системы жертвы путем наблюдения за косвенными признаками ее физической активности; не подразумевает поиск свежих уязвимостей или подбор эксплойтов. Атака по сторонним каналам включает в себя корреляционный анализ энергопотребления, акустический анализ и чтение остаточных данных после их удаления из памяти;
- **спуфинг (подмена)** – злоумышленник выдает себя за другого пользователя или подменяет устройство в сети;
- **SYN-флуд** – злоумышленник перехватывает и подделывает пакет TCP:SYN, отправленный компьютером. Таким образом создаются многочисленные полуоткрытые соединения с несуществующими адресами, из-за чего у компьютера могут закончиться ресурсы;
- **уязвимости нулевого дня** – дыры в безопасности или ошибки в коммерческом/промышленном ПО, неизвестные разработчикам или производителям.

Термины, связанные с защитой

Ниже перечислены термины и определения, относящиеся к разным киберзащитным механизмам и технологиям:

- **ASLR** – рандомизация размещения адресного пространства (англ. *Address Space Layout Randomization*) – это механизм защиты памяти и предотвращения атак на основе переполнения буфера. Он делает непредсказуемым выбор участка памяти для загрузки исполняемого файла. Вредонос, внедряющий свой код после переполнения буфера, не знает, куда он будет загружен, поэтому управление указателем на текущую инструкцию становится чрезвычайно сложным. ASLR также защищает от атаки возврата в библиотеку;
- **черная дыра** (или **воронка**) – при обнаружении DDoS-атаки создаются маршруты, которые направляют зловредные данные от DNS-сервера или

IP-адреса в никуда. Воронки производят дополнительный анализ, чтобы отфильтровать полезные данные;

- **предотвращение выполнения данных** – *DEP* (англ. *Data Execution Prevention*) маркирует области памяти как исполняемые и неисполняемые. Это не дает злоумышленнику выполнить код, внедренный в такие области в результате переполнения буфера. Результатом будет системная ошибка или исключение;
- **глубокий анализ пакетов** – *DPI* (англ. *Deep Packet Inspection*) анализирует каждый пакет (его тело и, возможно, заголовок) в потоке данных, чтобы изолировать инструкции, вирусы, спам и другую информацию, фильтруемую по определенным критериям;
- **брандмауэр** – механизм сетевой безопасности, который пропускает или блокирует потоки пакетов между доверенной и недоверенной зонами. Для контроля и управления трафиком на конкретных маршрутах можно использовать *списки контроля доступа* (англ. *access control lists,* или *ACL*). Брандмауэр может выполнять фильтрацию с сохранением состояния и соблюдать правила, основанные на целевых портах и состоянии трафика;
- **безопасные адресные пространства и неисполняемая память** – защищает неисполняемые участки памяти, доступные для записи. Защищает от NOP-сдвигов. На платформе Intel это бит NX, а в ARM – XN;
- **приманка** – инструмент для обнаружения, перенаправления или обратного анализа вредоносных атак. Приманка выглядит как обычный вебсайт или сетевой узел, но при этом она изолирована и подлежит тщательному мониторингу. Данные и запросы, поступающие в устройство, записываются в журнал;
- **управление доступом к памяти на основе инструкций** – методика отделения данных о возвращаемого адреса внутри стека. Помогает защититься от ROP-атак и особенно полезна в ограниченных IoT-системах;
- **система обнаружения вторжений** – *IDS* (англ. *Intrusion Detection System*) (сетевой механизм для обнаружения сетевых угроз за счет внеполосного анализа потока пакетов) не привязан к источнику или адресату, поэтому позволяет реагировать в режиме реального времени;
- **система предотвращения вторжений** – блокирует сетевые угрозы с помощью полноценного линейного анализа, статистических методов обнаружения и фильтрации по сигнатуре;
- **подставной ботнет** – инструмент, эмулирующий зараженное устройство. Подключается к управляющему серверу и принимает вредоносные команды, которые тот передает своему ботнету;
- **сканирование портов** – методика, позволяющая находить открытые и доступные порты в локальной сети;
- **инфраструктура открытых ключей** – *PKI* (англ. *Public Key Infrastructure*) определяет иерархические механизмы, которые гарантируют подлин-

ность происхождения открытого ключа. Сертификат подписывается удостоверяющим центром;

- **открытый ключ** – генерируется с помощью закрытого ключа и доступен для внешних клиентов. Открытый ключ может использоваться для расшифровки хешей;
- **закрытый ключ** – генерируется с помощью открытого ключа и никогда не выставляется наружу. Хранится в безопасном месте и используется для шифрования хешей;
- **корень доверия** – *RoT* (англ. *Root of Trust*) запускается в самом начале загрузки устройства и находится в неизменяемом, доверенном участке памяти (таком как ROM). Если BIOS или программа начальной загрузки доступны для неконтролируемого изменения, корень доверия теряет всякий смысл. RoT обычно является первой ступенью многоэтапной безопасной загрузки;
- **безопасная загрузка** – последовательность этапов загрузки устройства. Начинается с корня доверия и проходит через запуск ОС и приложений; каждый компонент должен иметь подлинную подпись. Проверка подписей производится с помощью открытых ключей, загруженных на предыдущих этапах;
- **стековые индикаторы** – защищают стек от переполнения и предотвращают выполнение кода, размещенного в стеке;
- **безопасная среда выполнения** – безопасный участок процессора, код и данные в котором защищены. Обычно эта среда выполнения находится в ядре основного процессора и следит за тем, чтобы безопасная загрузка, денежные транзакции и работа с закрытыми ключами выполнялись на более высоком уровне безопасности по сравнению с обычным кодом.

Анатомия кибератак на IoT-устройства

Область кибербезопасности слишком широкая, чтобы ее можно было втиснуть в одну главу. Но, поскольку интернет вещей состоит из аппаратного обеспечения, сети, протоколов, сигналов, облачных компонентов, фреймворков, операционных систем и всего, что их связывает, все эксплойты и атаки на IoT-устройства можно разделить на три основных типа:

- **Mirai** – самая разрушительная DDoS-атака в истории, спровоцированная плохой защитой IoT-устройств в отдаленных районах;
- **Stuxnet** – правительственное кибероружие, нацеленное на IoT-устройства в SCADA-системах. Нанесло значительный и непоправимый ущерб ядерной программе Ирана;
- **цепная реакция (Chain Reaction)** – исследовательская методика для эксплуатации личных сетей. Использует устройства вроде «умных» лампочек и не требует подключения к интернету.

Имея представление о том, как работают эти угрозы, архитектор может подобрать технологии для их предотвращения и наладить процессы, которые позволят нивелировать ущерб от аналогичных атак.

Mirai

Mirai – это название вредоноса, который в августе 2016 г. заразил IoT-устройства на базе Linux. Атака выполнялась из ботнета, сгенерировавшего громадную DDoS-нагрузку. В число самых известных жертв вошли Krebs on Security (популярный блог о безопасности), Dyn (очень популярный и востребованный DNS-провайдер)и Lonestar (крупный сотовый оператор в Либерии). Среди менее значимых целей были итальянские политические сайты, серверы *Minecraft* в Бразилии и российские интернет-аукционы. Косвенными жертвами DDoS-атаки на компанию Dyn стали ее клиенты, среди которых оказались такие огромные сервисы как PlayStation Network, Amazon, GitHub, Netflix, PayPal, Reddit и Twitter. Всего было заражено 600 000 устройств, которые стали частью ботнета.

Исходный код Mirai был опубликован на хакерском форуме **hackforums.net**. Путем его анализа, а также с помощью трассировки и изучения журнальных файлов исследователи раскрыли принцип работы и хронологию выполнения атаки:

- **поиск жертв** – быстрое асинхронное сканирование с использованием пакетов TCP SYN и проверкой произвольных IPV4-адресов. Особое внимание вредонос уделял TCP-порту 23, принадлежащему SSH/Telnet, и порту 2323. В случае нахождения подходящего порта начинался второй этап. В Mirai был встроен черный список адресов, которые следует избегать. Он содержал 3,4 миллиона записей, принадлежащих почтовой службе США, Hewlett Packard, GE и американскому министерству обороны. Скорость сканирования достигала 250 б/с. Для ботнета это довольно медленно. Атаки вроде SQL Slammer генерировали сканирующие запросы на скорости 1,5 Мбит/с. Дело в том, что IoT-устройства имеют куда более скромные вычислительные возможности по сравнению с настольными и мобильными компьютерами;
- **простой перебор через Telnet** – на этом этапе вредонос пытался установить рабочее соединение с жертвой через Telnet, отправляя 10 пар «логин-пароль», выбранных случайным образом из списка с 62 парами. В случае успеха взломанный компьютер подключался к серверу C2. Более поздние разновидности Mirai научились выполнять RCE-эксплойты;
- **заражение** – сервер передавал потенциальной жертве программу-загрузчик, которая отвечала за определение версии операционной системы и установку вредоноса, заточенного под конкретное устройство. Затем загрузчик искал и прекращал работу конкурирующих процессов, которые использовали порты 22 или 23 (в том числе и другие вредоносы, которые могли находиться на устройстве). После этого загрузчик удалялся, а имя процесса маскировалось, чтобы скрыть его присутствие. Вредо-

нос не хранился на постоянном накопителе и исчезал после перезагрузки. В конце бот входил в спящий режим в ожидании дальнейших команд.

Жертвами Mirai были такие IoT-устройства, как IP-камеры, DVR, потребительские маршрутизаторы, IP-телефония, принтеры и цифровые телевизионные приставки. Вредоносные двоичные файлы поддерживали 32-битные версии ARM, MIPS и x86.

1 августа 2016 г. произошло первое сканирование, которое было выполнено с американского веб-хостинга. На поиск первого компьютера с открытым портом и подходящим паролем ушло 120 минут. Минуту спустя было заражено еще 834 устройства. Через 20 часов количество зараженных устройств достигло 64 500. Еще через 75 минут охват Mirai удвоился. Большинство зараженных компьютеров, ставших частью ботнета, находились в Бразилии (15%), Колумбии (14%) и Вьетнаме (12,5%), хотя DDoS-атаки были нацелены на другие регионы.

Ущерб был ограничен DDoS-атаками, которые проводились через SYN-пакеты, сети GRE IP, а также протоколы STOMP и DNS. На протяжении пяти месяцев серверы C2 передали 15 194 отдельные команды, направленные против 5042 веб-сайтов. 21 сентября 2016 г. ботнет Mirai произвел массированную DDoS-атаку на Krebs (блог, посвященный безопасности), сгенерировав трафик на скорости 623 Гбит/с. Это была худшая DDoS-атака в истории. Ниже показан снимок экрана, сделанный во время этих событий с помощью сервиса www.digitalattackmap.com, который стал результатом сотрудничества между NETSCOUT и Google Jigsaw (рис. 12.1).

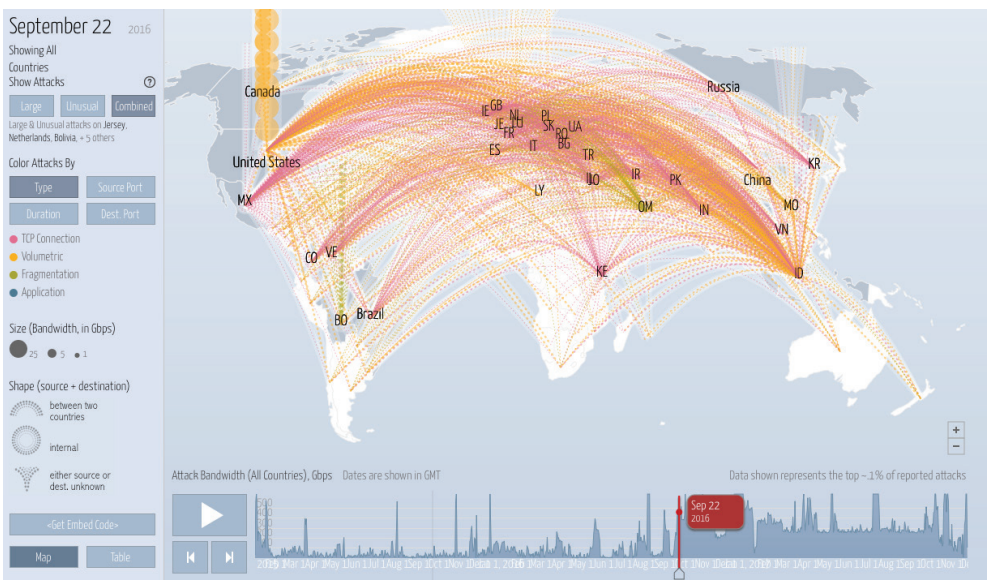


Рис. 12.1 ❖ Схема DDoS-атаки Mirai на блог Krebs, любезно предоставленная сервисом www.digitalattackmap.com

Stuxnet

Stuxnet – это первое задокументированное кибероружие, предназначенное для нанесения урона активам другой страны. Это был червь, направленный против *программируемых логических контроллеров* (англ. *Programmable Logic Controllers, или PLC*) Siemens, основанных на SCADA. Он использовал руткит для изменения скорости вращения двигателей, находящихся под непосредственным управлением PLC. Создатели этого вируса сделали все для того, чтобы он атаковал только устройства с ведомыми частотно-регулируемыми приводами, которые подключены к модулям Siemens S7-300 PLC и вращаются с частотой 807 Гц или 1210 Гц, так как они обычно применяются в насосах и газовых центрифугах для обогащения урана.

Атака предположительно началась в апреле или марте 2010 г. Процесс заражения состоял из следующих этапов:

1) **начальное заражение** – вначале червь заразил компьютер с Windows, используя уязвимости, обнаруженные в предыдущих вирусных атаках. Считается, что это было сделано через подключение USB-диска. При этом использовалось сразу несколько эксплойтов нулевого дня (беспрецедентная изощренность). Эксплойты запускали руткит в режиме пользователя и ядра, а затем устанавливали драйвер устройства с корректным сертификатом, похищенный у компании Realtek. Драйвер, работающий в режиме ядра, был необходим для скрытия Stuxnet от различных антивирусных пакетов;
2) **атака на Windows и распространение** – после установки через руткит вирус начинал искать в системе Windows файлы, связанные с контроллером Siemens SCADA версии WinCC/PCS 7, также известным как Step-7. В случае успеха червь пытался подключиться к серверу C2 через интернет, используя подставные URL-адреса (www.mypremierfutbol.com и www.todaysfutbol.com), чтобы обновиться до последней версии. Затем он искал на диске файл с названием **s7otbdx.dll**, который представлял собой важную коммуникационную библиотеку для взаимодействия между Windows и PLC. Контроллер Step-7 включал в себя встроенную базу данных с паролями, которая взламывалась с помощью еще одного эксплойта нулевого дня. Stuxnet внедрялся между системой WinCC и **s7otbdx.dll**, выполняя атаку посредника. Первым делом вирус записывал информацию о нормальном режиме работы центрифуг;
3) **разрушение** – когда было решено скоординировать атаку, вирус воспроизвел предварительно записанные данные и отправил их контроллерам SCADA, которые не подозревали о том, что система была скомпрометирована или ведет себя необычно. Ущерб был нанесен в ходе двух разных скоординированных атак, основанных на манипуляции PLC, в результате чего пострадал весь комплекс обогащению урана. Каждые 27 дней роторы центрифуги запускались на 15 или 50 минут, что приводило к их постепенному изнашиванию и появлению трещин в их шахтах. Кроме того, нарушался процесс обогащения.

Считается, что данная атака, произведенная на главный иранский объект по обогащению урана в городе Нетенз, вывела из строя более тысячи центрифуг. С тех пор код Stuxnet стал общедоступным и превратился в своеобразную игровую площадку для создания производных эксплойтов (github.com/micrictor/stuxnet).

Цепная реакция

Цепная реакция (англ. *Chain Reaction*) – это академическое исследование нового вида кибератак, нацеленных на персональные mesh-сети без подключения к интернету. Оно также показывает, насколько уязвимыми могут быть удаленные IoT-датчики и управляющие системы. Целью оригинальной атаки были лампочки Philips Hue, которые можно найти в так называемых умных домах с поддержкой управления через интернет или мобильные приложения. Эксплойт способен масштабироваться для атаки на целые умные города – достаточно лишь вкрутить одну зараженную лампочку.

Лампочки Philips Hue поднимают mesh-сеть на основе протокола Zigbee, который является частью инициативы *Zigbee Light Link* (*ZLL*) по обеспечению совместимости между разными методами освещения. ZLL-сообщения не шифруются и не подписываются, хотя для защиты ключей, обмен которыми происходит при добавлении лампочки в mesh-сеть, применяется криптография. В итоге утек главный ключ, известный всем участникам консорциума ZLL. Кроме того, согласно стандарту ZLL, подключаемая лампочка должна находиться в непосредственной близости к инициатору, что не дает ему перехватить контроль над лампочками своих соседей. Протокол Zigbee также предоставляет бесконтактный метод перепрограммирования устройств, хотя пакеты с прошивкой являются зашифрованными и имеют цифровую подпись.

План атаки, которую провели исследователи, состоял из четырех этапов:

1) взломать шифрование и цифровую подпись пакета с прошивкой;
2) написать и доставить зараженное обновление прошивки на одну единственную лампочку, используя взломанное шифрование и ключи;
3) скомпрометированная лампочка присоединится к сети с помощью главного ключа, похищенного ранее, и обойдет защиту на основе близости размещения, используя дефект нулевого дня, найденный в широко распространенном компоненте Atmel AtMega;
4) после успешного подключения к mesh-сети Zigbee вредоносный код будет разослан соседним лампочкам, что приведет к их быстрому заражению. Распространение вируса будет происходить в соответствии с *теорией перколяции* и охватит все осветительные приборы в городе.

Для шифрования бесконтактных обновлений прошивки Zigbee использует AES-CCM (часть стандарта IEEE 802.15.4, о котором мы поговорим позже в этой главе). Для его взлома злоумышленник использует *корреляционный* и *дифференциальный анализ энергопотребления* (англ. *Correlation Power Analysis* [*CPA*] и *Differential Power Analysis* [*DPA*]). Это изощренный вид атаки, который под-

разумевает помещение лампочки на специальный стенд и измерение энергии, которую она потребляет. Учитывая развитые средства управления, можно измерить динамическое энергопотребление процессора, который выполняет инструкцию или перемещает данные (например, во время работы алгоритма шифрования). Это простой анализ энергопотребления, который оставляет мало шансов на взлом ключа. Методы CPA и DPA являются более продвинутыми и используют статистическую корреляцию. Вместо попытки распознать отдельные биты, CPA может оперировать целыми байтами. Показатели питания снимаются с помощью осциллографа и разбиваются на два множества в зависимости от промежуточного значения, которое взламывается: в первом оно равно 1, а во втором – 0. Реальное значение вычисляется путем вычитания из этих множеств среднего показателя.

Путем использования DPA и CPA исследователям удалось взломать систему освещения Philips Hue:

- метод CPA применялся для взлома AES-CBC. У атакующей стороны не было ни ключа, ни случайного числа, ни вектора инициализации. Благодаря этому подходу был получен ключ, который затем с помощью того же метода использовался для взлома случайного числа;
- метод DPA применялся для взлома режима счетчика AES-CTR и впоследствии – алгоритма шифрования, который использовался при упаковке прошивки. Исследователи обнаружили 10 участков, в которых, предположительно, выполнялся режим AES-CTR, что оставляло 10 потенциальных решений;
- затем исследователи сосредоточились на взломе защиты Zigbee, основанной на близости размещения, чтобы подключиться к сети. В результате изучения исходного кода загрузчика, который использовался в чипе Atmel, была найдена уязвимость нулевого дня. В частности, в момент отправки запроса сканирования в Zigbee проверка на удаленность проходила успешно. Чтобы ее обойти, достаточно было начать сеанс с любого другого сообщения. Это позволило исследователям подключиться к сети.

Настоящая атака могла бы заставить взломанную лампочку заразить своих соседей в радиусе ста метров, передав им вредоносный код для отключения обновлений, чтобы их невозможно было восстановить. В сущности, лампочки были бы под управлением злоумышленника, и их пришлось бы уничтожить. Исследователи сумели создать полностью автоматизированную систему для атаки и прикрепить ее к беспилотному летательному аппарату, который делал систематический облет в радиусе приема лампочек Philips Hue на территории университета и заражал каждую из них.

Больше информации об атаке на Zigbee методом CPA можно найти в докладе Э. Ронена, А. Шамира, А. О. Вейнгартена и К. О'Флинна «IoT Goes Nuclear: Creating a ZigBee Chain Reaction» (стр. 195–212), подготовленном к симпозиуму по безопасности и конфиденциальности IEEE (Сан-Хосе, 2017 г.). Отличное руководство и исходный код для проведения атаки CPA можно найти на вики-странице ChipWhisperer: wiki.newae.com/AES-CCM_Attack.

Физическая и аппаратная безопасность

Многие IoT-устройства находятся в удаленных и изолированных районах, что оставляет уязвимыми датчики и пограничные маршрутизаторы. Аппаратное обеспечение тоже требует современных механизмов защиты, которые широко используются в процессорах и микросхемах мобильных и других потребительских устройств.

Корень доверия

Первый уровень аппаратной безопасности заключается в установлении *корня доверия* (англ. *Root of Trust,* или *RoT*). RoT – это процесс загрузки с аппаратной проверкой подлинности, который гарантирует, что источник первой исполняемой инструкции не подлежит изменению. Это ключевой этап процесса загрузки, который участвует в дальнейшем запуске системы – от BIOS до ОС и приложений. RoT является базовой защитой от руткитов.

Каждый этап процесса загрузки проверяет подлинность следующего этапа, формируя таким образом *цепочку доверия*. Корень доверия может использовать разные методы запуска:

- загрузка образа и корневого ключа из прошивки или неизменяемой памяти;
- хранение корневого ключа в одноразовой программируемой памяти с помощью фьюз-битов;
- загрузка кода из защищенной области памяти в защищенное хранилище.

Корень доверия должен проверять подлинность каждого последующего этапа загрузки. Для этого на каждом этапе используется набор ключей с цифровой подписью (рис. 12.2).

В разных процессорах корень доверия реализован по-разному. Intel и ARM поддерживают следующие технологии:

- **ARM TrustZone** – ARM продает производителям чипов проприетарный кремниевый блок, который предоставляет корень доверия и другие механизмы безопасности. TrustZone делит аппаратные компоненты на безопасные и небезопасные. Таким образом микропроцессор отделяется от небезопасного ядра; он выполняет Trusted OS – защищенную операционную систему с четко определенным интерфейсом взаимодействия с небезопасными компонентами. Защищенные ресурсы и функции находятся в доверенном ядре и должны быть как можно более легковесными. Переход между компонентами разного типа делается с помощью аппаратного переключения контекста, благодаря чему отпадает необходимость в безопасном ПО для мониторинга. TrustZone также используется для управления системными ключами, денежными транзакциями и защитой авторских прав. Поставщикам оборудования доступно два профиля: A («application») и M («microcontroller»). Сочетания корня доверия, Trusted OS и этого типа процессоров называется *доверенной средой выполнения* (англ. *Trusted Execution Environment,* или *TEE*);

- **Intel Boot Guard** – это аппаратный механизм для проверки подлинности начального блока загрузки криптографическими средствами или с помощью процесса измерения. Для проверки начального блока производитель должен сгенерировать 2048-битный ключ, который состоит из двух частей: открытой и закрытой. Открытый ключ печатается на плате путем «детонации» фьюз-битов на этапе производства. Эти биты являются одноразовыми и не подлежат изменению. Закрытая часть ключа генерирует цифровую подпись для последующего удостоверения подлинности этапа загрузки.

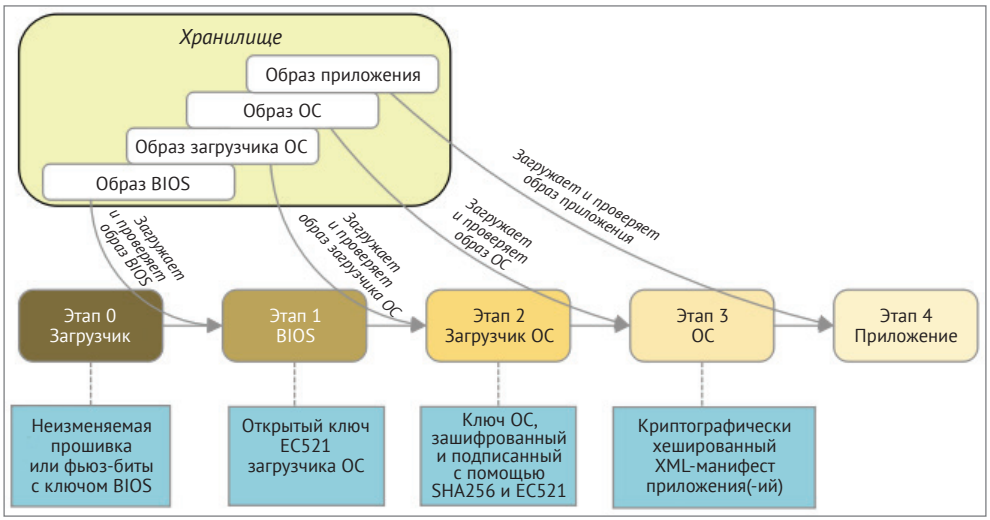


Рис. 12.2 ❖ Установление корня доверия. Выше представлена пятиэтапная загрузка, которая формирует цепочку доверия и начинается с загрузчика, находящегося в неизменяемой памяти. На каждом этапе используется открытый ключ, с помощью которого удостоверяется подлинность следующего загружаемого компонента

Управление ключами и модули TPM

Открытые и закрытые ключи являются залогом безопасной системы. Для их защиты требуется надлежащий механизм управления. Одним из самых популярных стандартов аппаратной защиты ключей является *TPM* (англ. *Trusted Platform Module* – доверенный платформенный модуль). Его спецификация была создана консорциумом Trusted Computing Group и является частью ISO и IEC. Текущая версия TMP 2.0 была выпущена в сентябре 2016 г. Оборудование, поставляемое Министерству обороны США, должно поддерживать TPM 1.2.

TPM представляет собой отдельный аппаратный компонент с RSA-ключом, вшитым на этапе производства.

Обычно TPM используется для хранения, защиты и администрирования ключей в таких сценариях, как шифрование диска, загрузка корня доверия, проверка

подлинности оборудования и программного обеспечения, а также для управления паролями. TPM может создать хеш проверенной аппаратной или программной конфигурации, который поможет обнаружить стороннее вмешательство на этапе выполнения. Эта технология также применяется в создании хешей типа SHA-1 и SHA-256, шифровании блоков методом AES, ассиметричном шифровании и генерации случайных чисел. Производством TPM-устройств занимаются такие компании как Broadcom, Nation Semiconductor и Texas Instruments.

Адресное пространство в процессоре и памяти

Мы уже обсудили различные эксплойты и технологии процессора, которые им противостоят. Двумя основными механизмами защиты в ЦПУ и ОС, на которые следует обратить внимание, являются неисполняемая память и рандомизация размещения адресного пространства. Обе они предназначены для усложнения или предотвращения процесса внедрения вредоносного кода на основе переполнения буфера или стека:

- **неисполняемая память** – это аппаратный механизм, с помощью которого операционная система делает участки памяти неисполняемыми. Конечная цель состоит в том, чтобы выполняться могли только те области памяти, в которых находится проверенных и подлинный код. При попытке внедрения вредоноса через переполнение стека система пометит соответствующий участок как неисполняемый, в результате чего сдвиг указателя текущей инструкции к этому участку приведет к аппаратному исключению. Маркировка неисполняемой памяти делается с помощью бита NX (через буфер ассоциативной трансляции). На платформах Intel и ARM этот бит называется XD (англ. *eXecute Disable* – выключение выполнения) и, соответственно, XN (англ. *eXecute Never* – никогда не выполнять). Эта технология поддерживается в большинстве ОС, таких как Linux и Windows, а также в некоторых системах реального времени;
- **рандомизация размещения адресного пространства** – ASLR, скорее, является особенностью работы с виртуальным адресным пространством в операционной системе, нежели аппаратной функцией, но ее тоже важно рассмотреть. Эта технология защищает от переполнения буфера и атаки возврата в библиотеку. Подобные методы взлома требуют от злоумышленника понимания структуры памяти и заключаются в намеренном выполнении определенного доброкачественного кода или библиотек. Это непростая задача, особенно если адресное пространство изменяется случайным образом при каждой загрузке. В Linux поддерживается с помощью заплаток PAX и Exec Shield. Microsoft тоже предоставляет защиту для кучи, стека и блоков обработки.

Безопасность хранения данных

Многие IoT-устройства используют постоянное хранилище на пограничном узле или маршрутизаторе/шлюзе. Умным туманным узлам (англ. *fog nodes*) тоже

нужно где-то хранить свои данные. Безопасность данных является ключевым аспектом предотвращения установки вредоносного ПО и защиты конфиденциальной информации в случае похищения устройства. Многие хранилища, такие как flash-накопители и жесткие диски, поддерживают шифрование и защитные технологии.

FIPS 140-2 (федеральный стандарт обработки информации) – это правовая норма, описывающая требования к шифрованию и безопасности для электронных устройств, хранящих конфиденциальные данные. Помимо технических требований она определяет правила и процедуры. FIPS 140-2 предусматривает несколько уровней безопасности:

- **уровень 1** – сугубо программное шифрование. Ограниченная безопасность;
- **уровень 2** – обязательная аутентификация на основе ролей и способность обнаруживать физическое проникновение с помощью специальных пломб;
- **уровень 3** – предусматривает устойчивость к физическому взлому. При попытке проникновения устройство удалит важные параметры безопасности. Включает в себя криптографическую защиту, управление ключами и аутентификацию с проверкой подлинности;
- **уровень 4** – продвинутая защита от взлома для продуктов, предназначенных для работы в физически незащищенной среде.

Помимо шифрования, также следует позаботиться о безопасности накопителей, выводимых из эксплуатации. Извлечение содержимого из старых систем хранения данных – относительно простая задача. Существуют дополнительные стандарты, описывающие безопасный процесс удаления данных с накопителя (будь то диск с магнитными пластинами или память с изменением фазового состояния). Кроме того, лаборатория NIST публикует документы о безопасном уничтожении содержимого, такие как «NIST Special Publication 800-88 for Secure Erase».

Физическая безопасность

Устойчивость к проникновению и физическая безопасность играют важную роль в интернете вещей. Многие IoT-устройства размещены удаленно, без всякой защиты. Это напоминает историю с проектом «Энигма» во время Второй мировой войны. Извлечение рабочей шифровальной машины из немецкой подводной лодки U-110 помогло взломать шифр. Злоумышленник с непосредственным доступом к IoT-устройству может использовать любые инструменты для взлома системы, как мы видели на примере эксплойта «Цепная реакция».

Ранее уже был представлен пример атаки сторонними каналами с помощью анализа энергопотребления; взлом также может осуществляться на основе времени, кэша, излучения электромагнитного поля и цепочки сканирования. Главная особенность атак сторонними каналами состоит в том, что взломанное устройство, по сути, превращается в тестовую площадку. Это означает, что оно будет находиться под наблюдением в контролируемой среде и его активность будет всячески измеряться.

Кроме того, такие методики, как DPA, используют статистический анализ для выведения закономерности между случайным вводом и выводом. Этот

подход применим только в случае, если система демонстрирует идентичное поведение с одним и тем же вводом (табл. 12.1).

Таблица 12.1. Методики выявления атак

	Методология
Атаки на основе времени	Попытка использования небольшой разницы во времени в работе алгоритма. Например, при замере времени работы алгоритма декодирования пароля можно заметить, что процедура завершается раньше обычного. Злоумышленник может также отслеживать использование кэша, которое характеризует алгоритм
Простой анализ энергопотребления	Этот подход похож на атаку по времени. Он заключается в измерении колебаний тока в зависимости от поведения алгоритма и выполнения машинных инструкций. Особенно уязвимыми являются открытые ключи. Для выполнения анализа требуется лишь несколько замеров, но показания должны иметь высокую точность. Большинство алгоритмов активно используют математические операции, и при этом разные инструкции демонстрируют разные шаблоны энергопотребления
Дифференциальный анализ энергопотребления	DPA измеряет динамическое энергопотребление, но может засечь изменения, которые были бы слишком незначительными при простом анализе. Передавая на вход системе случайные данные (например, разные произвольные ключи), злоумышленник может произвести тысячи измерений и выстроить закономерность. Например, для взлома алгоритма AES достаточно сформировать два набора результатов, которые зависят от входящего значения (0 или 1). Каждый набор сводится к среднему показателю, а разница между ними демонстрирует влияние случайного ввода на итоговый вывод

Методы предотвращения этих атак хорошо известны, их можно лицензировать и использовать в разного рода оборудовании. Среди контрмер можно выделить следующие:

- изменение функции шифрования, чтобы минимизировать использование ключа. Использование ключей, действительных только на время текущего сеанса и основанных на хеше оригинального ключа;
- для атак по времени: случайная вставка функций, которые не нарушают работу алгоритма; использование случайных машинных инструкций для создания большой рабочей функции, которую сложно взломать;
- удаление условных ответвлений, которые зависят от ключа;
- для атак на основе энергопотребления: минимизация утечек и ограничение операций с ключом. Это уменьшит рабочий набор показателей злоумышленника;
- внедрение помех в линии электропередач. Варьирование времени выполнения операций или смещение таймеров;
- изменение порядка следования независимых операций. Это снижает степень корреляции вокруг вычислений в S-блоке.

Другие аспекты, связанные с аппаратным обеспечением: предотвращение доступа к отладочным портам и каналам, которые часто представлены в виде последовательных портов и JTAG-портов. Чтобы совсем исключить отладочный доступ, следует убрать заголовки и детонировать фьюз-биты.

Модули ASIC обычно используют выводы типа BGA (англ. *ball grid array* – массив шариков) для атаки на PCA. В плате должен использоваться прочный, термостойкий клей, чтобы попытка физического взлома приводила к непоправимым повреждениям.

Криптография

Шифрование и секретность являются обязательными для IoT-устройств. Они помогают обезопасить взаимодействие, защищая прошивку и процесс аутентификации. Шифрование можно разделить на три основные категории (рис. 12.3):

- **симметричное шифрование** – для шифрования и дешифрования применяется один и тот же ключ. Симметричными являются такие алгоритмы как RC5, DES, 3DES и AES;
- **шифрование с открытым ключом** – ключ, использованный для шифрования данных, доступен публично. Но только принимающая сторона владеет закрытым ключом для расшифровки сообщения. Такой вид

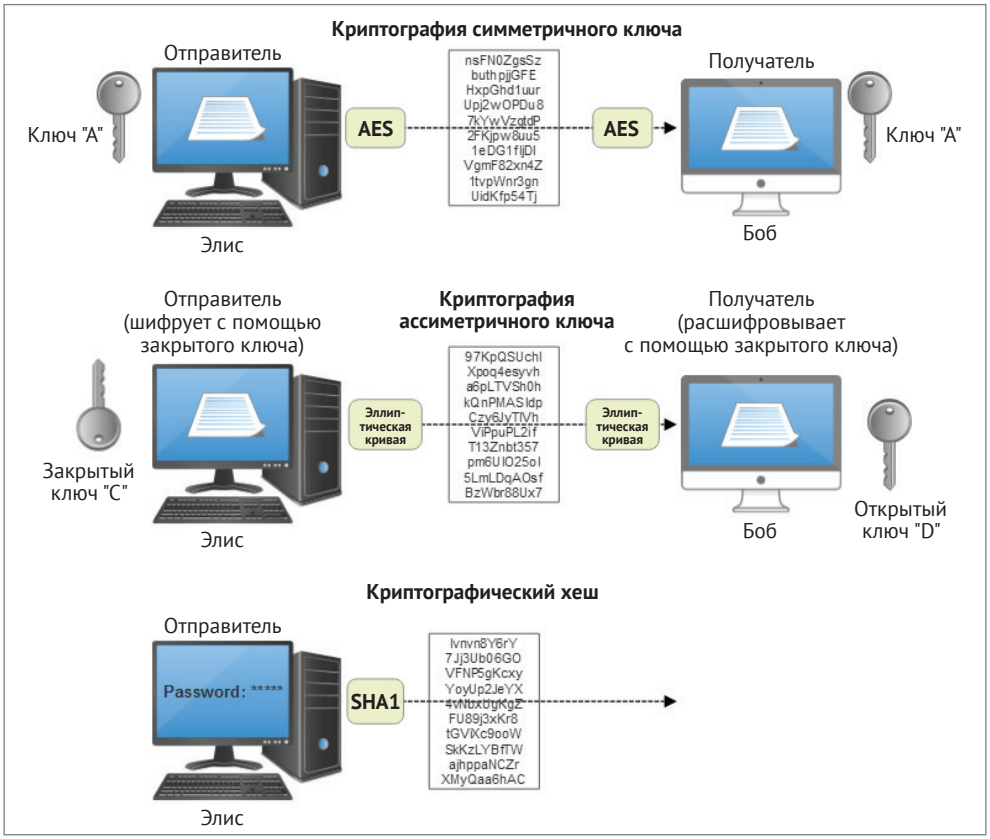


Рис. 12.3 ❖ Элементы криптографии. Здесь представлены симметричная, ассиметричная и хеширующая функции. Обратите внимание на использование ключей в первых двух. Симметричный алгоритм требует использования идентичных ключей для шифрования и расшифровки данных. Он выигрывает по скорости у симметричного шифрования, но его ключи должны храниться в безопасном месте

шифрования также называют ассиметричным. Ассиметричная криптография используется для обеспечения секретности данных, аутентификации и неотказуемости. Открытые ключи применяются в широко известных интернет-протоколах для шифрования и обмена сообщениями, таких как Elliptic Curve, PGP, RSA, TLS и S/MIME;

- **криптографическое хеширование** – привязывает данные произвольного размера к битовой строке (которую называют дайджестом). Хеш-функция с самого начала создается «однонаправленной». В сущности, единственный способ воссоздать итоговый хеш – перепробовать все возможные входящие комбинации (хеш-функцию нельзя выполнить в обратном направлении). Примерами однонаправленных хешей являются MD5, SHA1, SHA2 и SHA3. Обычно они применяются для шифрования цифровых подписей в образах прошивок, *имитовставках* и при аутентификации. В ходе шифрования небольших строк, таких как пароль, ввод может оказаться слишком коротким для создания полноценного хеша; в этом случае к паролю добавляется *соль* или публичная строка, чтобы увеличить энтропию. Соль – это разновидность *функции формирования ключа* (англ. *key derivation function,* или *KDF*).

Симметричная криптография

В криптографии используются такие термины, как *простой текст* (англ. *plaintext*) и *шифротекст* (англ. *ciphertext*), которые обозначают незашифрованный ввод и, соответственно, зашифрованный вывод. Текущим стандартом шифрования считается *AES* (англ. *Advanced Encryption Standard* – продвинутый стандарт шифрования); он пришел на смену старому алгоритму DES, разработанному в 1970-х гг. AES является частью спецификации FIPS и стандарта ISO/IEC 18033-3, которые используются во всем мире. Алгоритмы AES основаны на блоках фиксированной длины по 128, 192 или 256 бит. Сообщения, превышающие длину блока, разбиваются на несколько частей. AES состоит из четырех основных этапов шифрования. Ниже показан псевдокод для шифрования этим методом:

```
// Псевдокод для шифра AES-128
// in: 128 бит (простой текст)
// out: 128 бит (шифротекст)
// w: 44 слова по 32 бита каждое (расширенный ключ)
state = in

w=KeyExpansion(key) // Этап расширения ключа (в сущности, шифрование самого ключа)
AddRoundKey(state, w[0, Nb-1]) // Начальная итерация

forround = 1 step 1 toNr-1 //128 бит = 10 итераций, 192 бита = 12 итераций, 256 бит =
14 итераций
 SubBytes(state)   // Обеспечивает нелинейность шифра
 ShiftRows(state)  // Предотвращает независимое шифрование столбцов,
                      которое может ослабить алгоритм
 MixColumns(state) // Преобразует каждый столбец и добавляет рассеивание в алгоритм
```

```
 AddRoundKey(state, w[round*Nb, (round+1)*Nb-1]) // Генерирует подключ и объединяет его
                                                 с переменной state
end for

SubBytes(state) // Заключительная итерация и очистка
ShiftRows(state)
AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])

out = state
```

Длина ключей в AES равна 128, 192 или 256 бит. В целом, чем длиннее ключ, тем лучше защита. Размер ключа пропорционален количеству циклов процессора, необходимых для шифрования или расшифровки блока: 128 бит требует 10 циклов, 192 бита – 12 циклов, а 256 бит – 14 циклов.

Блочные шифры представляют собой алгоритмы, которые основаны на симметричном ключе и обрабатывают данные в виде последовательных блоков. Современные шифры основаны на статье о промышленном шифровании, написанной Клодом Шенноном в 1949 г. Режим шифрования – это алгоритм, который описывает многократное применение блочного шифра для преобразования больших объемов данных, состоящих из множества блоков. Большинство современных шифров используют *вектор инициализации* (англ. *Initialization Vector,* или *IV*), благодаря которому один и тот же ввод каждый раз превращается в разный шифротекст. Алгоритм AES имеет несколько режимов работы:

- **режим простой замены (англ. *Electronic Codebook,* или *ECB*)** – это самый простой вид AES-шифрования; он применяется в сочетании с другими режимами для улучшения безопасности. Данные делятся на блоки, каждый из которых шифруется отдельно. Идентичные блоки дают один и тот же результат, что делает этот подход относительно ненадежным;
- **режим сцепления блоков (англ. *Cipher Block Chaining,* или *CBC*)** – перед шифрованием к простому тексту применяется исключающее ИЛИ с предыдущим зашифрованным блоком;
- **режим обратной связи по шифротексту (англ. *Cipher Block Chaining,* или *CFB*)** – похож на CBC, но формирует поток шифров (вывод предыдущего шифра служит вводом для следующего). CFB использует предыдущий зашифрованный блок, чтобы сгенерировать ввод для текущего шифра. Из-за этой зависимости CFB нельзя выполнять параллельно. Поточные шифры допускают потерю блока при передаче; в этом случае он будет восстановлен на основе последующих блоков;
- **режим обратной связи по выходу (англ. *Output Feedback Chaining,* или *OFB*)** – этот режим аналогичен CFB, но позволяет применять коды исправления ошибок еще до шифрования;
- **режим счетчика (англ. *Counter,* или *CTR*)** – превращает блочный шифр в поточный, используя инкрементальный счетчик, который распараллеливает подачу ввода каждому блочному шифру, что ускоряет выполнение. В качестве ввода используется сочетание счетчика и случайно сгенерированного числа;

- **CBC с имитовставкой (CBC-MAC)** – имитовставка (англ. *Message Authentication Code,* или *MAC*) используется для аутентификации сообщения и подтверждения того, что оно пришло от заявленного отправителя. Затем получатель добавляет имитовставку к сообщению для последующей проверки подлинности.

Эти режимы разрабатывались с конца 1970-х до начала 1980-х гг. и продвигались Национальным институтом стандартов и технологий в спецификации FIPS 8 в рамках алгоритма DES. Они обеспечивают конфиденциальность информации, но не защищают от ее изменения и подмены. В связи с этим начали использовать цифровые подписи, а сообщество специалистов по безопасности разработало режим CBC-MAC для аутентификации. Применение CBC-MAC в сочетании с исходными режимами было непростой задачей, пока не появились алгоритмы наподобие AES-CCM, которые предоставляют как аутентификацию, так и секретность. CCM расшифровывается как *Counter with CBC-MAC Mode* (счетчик с режимом CBC-MAC).

CCM – это важный режим, который используется для подписи и шифрования данных в целом ряде протоколов, рассмотренных в этой книге, включая Zigbee, Bluetooth Low Energy, TLS 1.2 (после обмена ключами), IPSEC и 802.11 Wi-Fi WPA2.

AES-CCM использует двойной шифр: CBC и CTR. AES-CTR (или режим счетчика) применяется для общей расшифровки входящего потока с шифротекстом, который содержит зашифрованную имитовставку. AES-CTR расшифровывает как имитовставку, так и сами данные. На этом этапе алгоритма формируется так называемая ожидаемая имитовставка; оригинальный заголовок фрейма и расшифрованные блоки, полученные на выходе из AES-CTR, помечаются как ввод. Данные расшифрованы, однако для аутентификации необходима имитовставка, вычисляемая в AES-CBC; если она отличается от той, которая ожидается на этапе AES-CTR, это означает, что данные могли быть изменены в процессе передачи.

На рис. 12.4 показан зашифрованный поток данных, который аутентифицируется с помощью AES-CBC и расшифровывается в режиме AES-CTR. Таким образом обеспечивается аутентификация и секретность оригинального сообщения.

Важным аспектом, с точки зрения IoT-устройств в полносвязной mesh-сети, является количество необходимых ключей. Для n узлов в mesh-сети с двунаправленным взаимодействием этот показатель равен $n(n-1)/2$ или $O(n^2)$.

Ассиметричная криптография

Ассиметричную криптографию также называют шифрованием с открытым ключом. Ассиметричные ключи генерируются попарно (для шифрования и дешифрования); они могут быть взаимозаменяемыми – то есть, один ключ может шифровать и расшифровывать, хотя это не обязательное требование. Но обыч-

но генерируется пара ключей – один открытый, а другой закрытый. В этом разделе описывается три основополагающих шифра с открытым ключом: RSA, протокол Диффи-Хеллмана и эллиптические кривые.

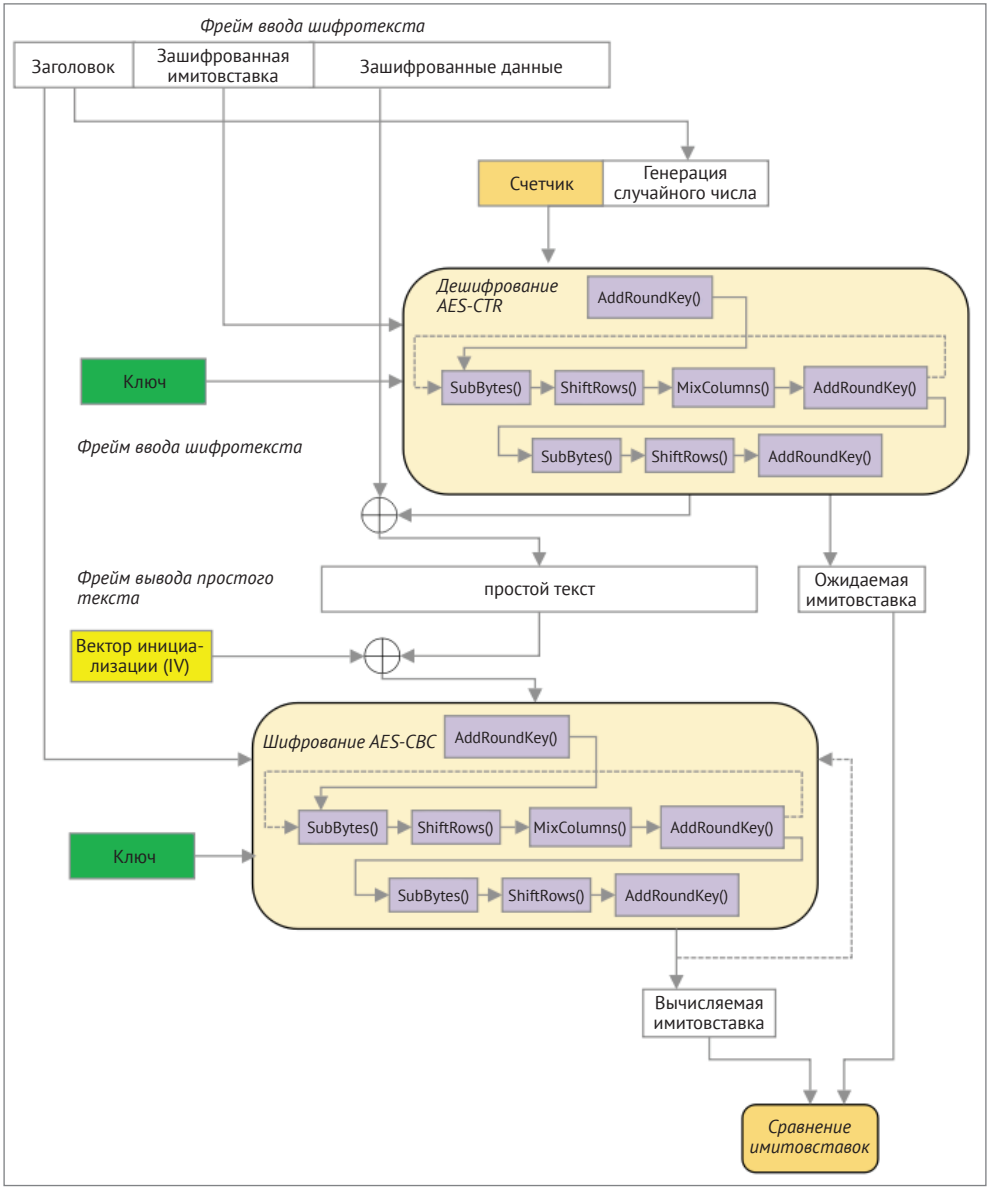


Рис. 12.4 ❖ Режим AES-CCM

Стоит отметить, что в отличие от симметричных ключей, количество которых вычисляется из расчета на взаимодействие любых двух узлов в mesh-сети, ассиметричная криптография требует лишь $2n$ или $O(n)$ ключей.

Первый метод ассиметричного шифрования с открытым ключом был описан в алгоритме *Ривеста-Шамира-Адлемана* (англ. *Rivest-Shamir-Adleman,* или *RSA*), разработанном в 1978 г. Он подразумевает, что пользователь должен найти и опубликовать произведение двух больших простых чисел и вспомогательное значение (открытый ключ). Открытый ключ позволяет шифровать сообщения и доступен кому угодно, но простые множители остаются конфиденциальными. Алгоритм выглядит так:

1) находим два больших простых числа, p и q;
2) $n = pq$;
3) $\varphi(n) = (p1)(q-1)$;
4) **открытый ключ** – выбираем целое число e, которое является взаимно простым для $\varphi(n)$ и находится в диапазоне $1 < e < \varphi(n)$; типичным значением является $2^{16} + 1 = 65537$;
5) **закрытый ключ** – вычисляем d для решения уравнения конгруэнции: $de \equiv 1(mod\ \varphi(n))$.

Таким образом, для шифрования сообщения используется открытый ключ (n, e), а для его расшифровки – закрытый ключ (n, d):

- **шифрование**: *шифротекст* = (*простой текст*)^e *mod n*;
- **расшифровка**: *простой текст* = (*шифротекст*)^d*mod n*.

Часто в короткие сообщения перед шифрованием добавляется сдвиг, чтобы получить хороший шифротекст.

Наверное, самым известным видом обмена ассиметричными ключами является протокол Диффи-Хеллмана (названный в честь Уитфилда Диффи и Мартина Хеллмана) (рис. 12.5). Типичным для ассиметричной криптографии считается понятие *односторонней функции с потайным входом* (англ. *Trapdoor Function*), которая принимает заданное значение A и возвращает вывод B; но при этом из B нельзя получить A.

Метод Диффи-Хеллмана позволяет обеим сторонам (Элис A и Боб B) обмениваться ключами, не зная заранее об общем ключе s. Алгоритм основан на открытом обмене начальным простым числом, p, и генераторе простых чисел g, который представляет собой *первообразный корень p*. Пусть закрытые ключи Элис и Боба называются a и b. Тогда $A = g^a\ mod\ p$ и $B = g^b\ mod\ p$. Свои секретные ключи Элис и Боб вычисляют как $s = B^a\ mod\ p$ и $s = B^a\ mod\ p$.

В итоге, $(g^a\ mod\ p)^b\ mod\ p = (g^b\ mod\ p)^a\ mod\ p$.

Сильной стороной такого обмена ключами является генерация по-настоящему случайного числа для каждого закрытого ключа. Малейшая предсказуемость в работе *генератора псевдослучайных чисел* может привести к взлому шифра. Однако принципиальным недостатком здесь выступает отсутствие аутентификации, что открывает возможность для MITM-атаки.

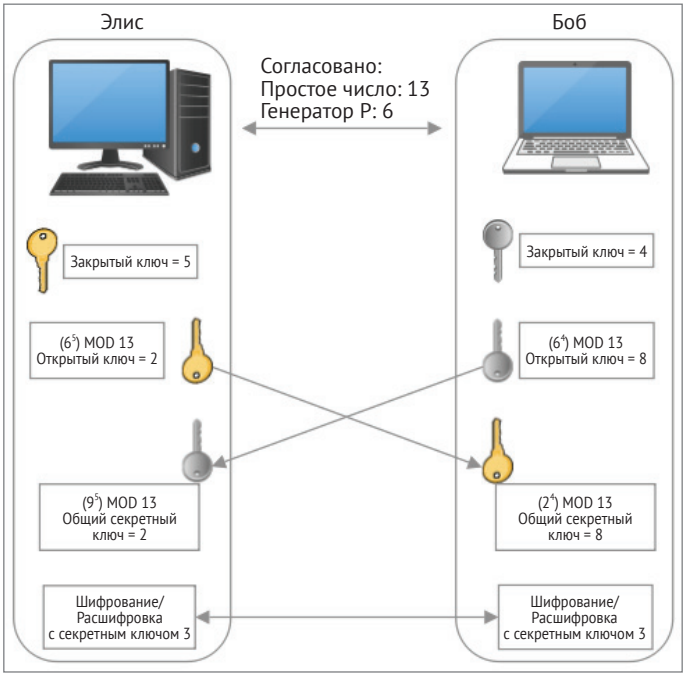


Рис. 12.5 ❖ Протокол Диффи-Хеллмана.
Процесс начинается с открытого обмена заранее согласованными простым числом и генератором простых чисел. Элис и Боб генерируют независимые закрытые ключи, а открытые ключи генерируются и отправляются по сети в незашифрованном виде. На основе этого получается секретный ключ, который используется для шифрования и расшифровки

Еще один способ обмена ключами, *протокол Диффи-Хеллмана на эллиптических кривых* (англ. *Elliptic-Curve Diffie-Hellman,* или *ECDH*), был предложен Коблитцем и Миллером в 1985 г. Он основан на алгебре эллиптических кривых над конечным полем. Алгоритм ECDH получил поддержку со стороны института NIST и был одобрен NSA для шифрования совершенно секретных материалов с использованием 384-битных ключей. *Криптография на основе эллиптических кривых* (англ. *Elliptic Curve Cryptography,* или *ECC*) обладает следующими отличительными характеристиками:

- симметричность по оси x;
- эллиптическая кривая может пересекаться с прямой линией не более чем в трех точках.

Процесс ECC (рис. 12.6) начинается с проведения прямой линии из заданной точки на грани в направлении *MAX*. Эта линия соединяет *A* и *B*. Скалярное произведение точки *A* используется для проведения линии между двумя точками, после чего на новом непомеченном пересечении чертится строго

вертикальная линия – либо вверх, либо вниз. Этот процесс повторяется *n* раз, где *n* – размер ключа.

Это похоже на конечный результат удара по бильярдному шару после того, как тот множество раз ударился о борта стола. Итоговое местоположение шара не позволяет наблюдателю определить, какой была его исходная позиция.

MAX – это максимальное значение по оси x, которое ограничивает удаленность вершины. Если вершина выходит за пределы *MAX*, алгоритм применяет это значение и устанавливает новую точку *x-MAX*, удаленную от исходной точки *A*. Значение *MAX* эквивалентно размеру используемого ключа. Длинный ключ генерирует больше вершин и повышает устойчивость к взлому. В сущности, это функция-обертка.

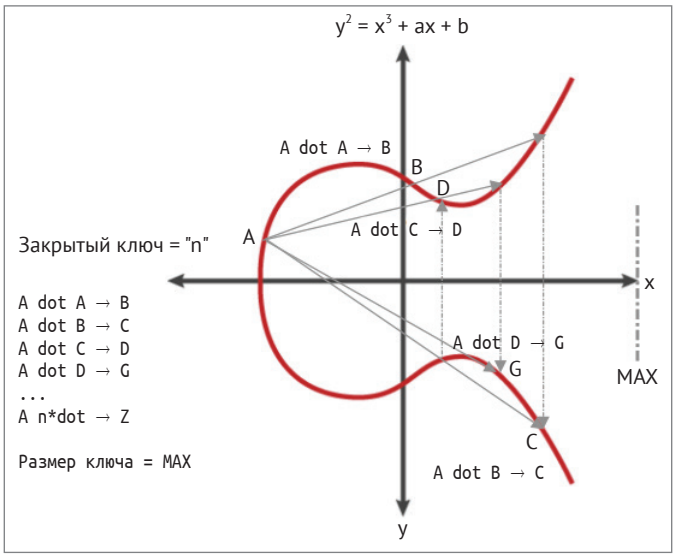


Рис. 12.6 ❖ Криптография на основе эллиптических кривых (ECC). Здесь показана стандартная эллиптическая кривая на осях x и y. Процесс начинается с проведения прямой линии из заданной точки A ко второй точке и поиска третьего, непомеченного пересечения. Линия проводится строго вертикально к противоположному значению по оси y, которое в этот момент маркируется. Процесс продолжается для n точек, соответствующих длине ключа

Эллиптические кривые начинают превалировать над RSA. Современные браузеры поддерживают алгоритм ECDH, который является предпочтительным методом аутентификации по SSL/TLS. Как вы позже увидите, ECDH можно найти в Bitcoin и некоторых других протоколах. На сегодняшний день RSA используется только в случае, если у SSL-сертификата есть подходящий RSA-ключ.

Еще одно преимущество состоит в том, что даже короткие ключи обеспечивают такую же криптографическую устойчивость, как и устаревшие методы. Например, 256-битный

ключ в ECC эквивалентен 3072-битному ключу в RSA. Это свойство следует учитывать в контексте ограниченных IoT-устройств.

Криптографический хеш (аутентификация и цифровая подпись)

Третьим видом технологии шифрования являются хеширующие функции. Обычно они используются для создания цифровых подписей и считаются «однонаправленными» – без возможности обратного выполнения. Для воссоздания исходных данных, прошедших через хеширующую функцию, пришлось бы перебирать все возможные комбинации ввода. Ключевые характеристики функции хеширования:

- всегда генерирует один и тот же хеш из одинакового ввода;
- быстрая в вычислении, но не мгновенная (см. доказательство выполнения работы);
- необратимая; не может сгенерировать исходное сообщение из хеша;
- малейшее изменение ввода вызывает существенную энтропию и полностью изменяет вывод;
- два разных сообщения никогда не будут иметь один и тот же хеш.

Принцип работы криптографических хеширующих функций, таких как SHA-1 (*Secure Hash Algorithm – алгоритм криптографического шифрования*), можно проиллюстрировать на примере двух строк, которые отличаются лишь одним символом:

```
Ввод: Boise Idaho
Хеш SHA1: 375941d3fb91836fb7c76e811d527d6c0a251ed4
Ввод: B0ise Idaho
Хеш SHA1: 182ae5b6e186a1b1857c7f18df84ae18438d0b57
```

Алгоритмы семейства SHA активно используются в:

- репозиториях Git;
- цифровых подписях TLS-сертификатов для веб-браузеров (HTTPS);
- проверке подлинности содержимого файла или образа диска.

Большинство хеш-функций основаны на структуре Меркла-Дамгарда. В приведенном ниже примере ввод разбивается на блоки одинакового размера, каждый из которых проходит через сжимающую функцию с применением результатов сжатия предыдущего блока. Для выбора начального значения используется вектор инициализации. Благодаря функции сжатия хеш получается устойчивым к коллизиям. Алгоритм SHA-1 построен на основе структуры Меркла-Дамгарда (рис. 12.7).

В целом, сообщения, которые подаются на ввод алгоритму SHA, должны быть меньше 264 бит. Они последовательно обрабатываются в 512-битных блоках. Стандарт SHA-1 вытеснен более устойчивыми версиями, такими как SHA-256 и SHA-3. В хешах SHA-1 нашли возможность «коллизий»; и хотя для этого требуется примерно от 2^{51} до 2^{57} операций, взлом хеша на арендованном графическом адаптере обойдется лишь в несколько тысяч долларов. В связи с этим рекомендуется перейти на другие разновидности SHA.

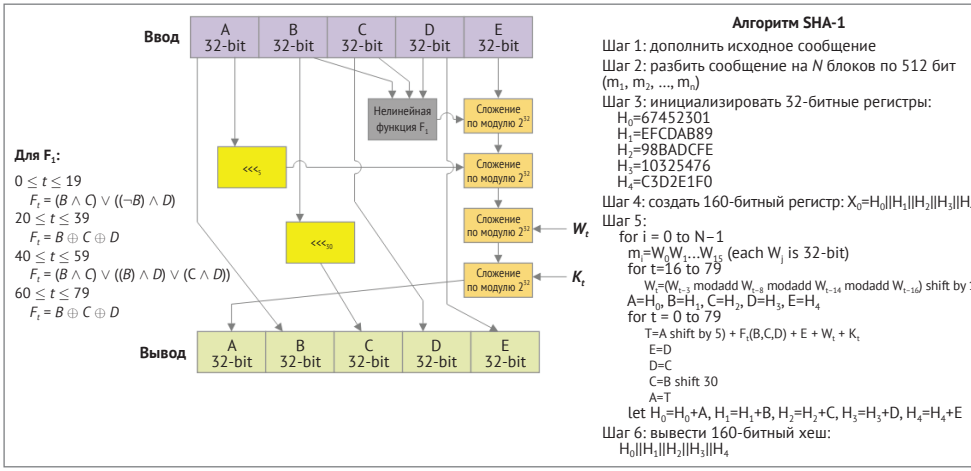


Рис. 12.7 ❖ Алгоритм SHA-1. Ввод разбивается на пять 32-битных блоков

Инфраструктура открытого ключа

Ассиметричная криптография (с открытым ключом) – это основа торговли и взаимодействия в интернете. Она повсеместно используется в SSL- и TLS-соединениях. Типичным примером является ситуация, когда передаваемые данные могут быть зашифрованы с помощью открытого ключа (то есть, кем угодно), но расшифровать их может тот, кому принадлежит закрытый ключ. Еще одно применение связано с цифровыми подписями, когда отправитель подписывает двоичные данные закрытым ключом, а получатель может проверить их подлинность, если у него есть открытый ключ.

Чтобы наладить надежную выдачу открытых ключей, используется процесс под названием *инфраструктура открытого ключа* (англ. *Public Key Infrastructure,* или *PKI*). Гарантия подлинности обеспечивается за счет *удостоверяющих центров* (англ. *Certificate Authorities,* или *CA*), которые управляют ролями и правилами, создавая распределенные цифровые сертификаты. Крупнейшими публичными издателями TLS-сертификатов являются компании Symantec, Comodo и GoDaddy. Форматы сертификатов на основе открытых ключей описываются стандартом X.509. Это основа безопасного взаимодействия в протоколах TLS/SSL и HTTPS. X.509 определяет такие атрибуты как используемый алгоритм шифрования, сроки годности и издатель сертификата.

В состав PKI входит *центр регистрации* (англ. *Registration Authority,* или *RA*), который аутентифицирует отправителя, управляет отдельными ролями/правилами и может отозвать сертификат. Для передачи списков отозванных сертификатов RA взаимодействует с *центром проверки подлинности* (англ. *Validation Authority,* или *VA*). CA выдает сертификат отправителю. Когда сообщение получено, VA может проверить подлинность ключа и удостовериться в том, что он не был аннулирован.

На рис. 12.8 показан пример инфраструктуры PKI. Здесь используются системы CA, RA и VA, а шифрование сообщения подразумевает выдачу ключа и проверку его подлинности.

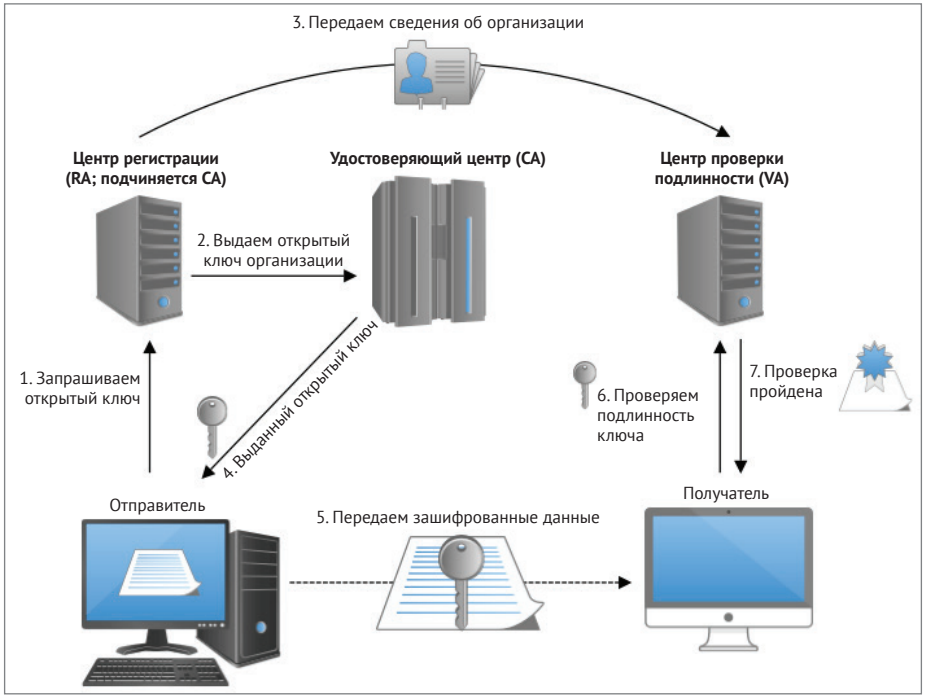


Рис. 12.8 ❖ Пример инфраструктуры PKI

Сетевой стек: протокол защиты транспортного уровня

Протокол защиты транспортного уровня (англ. *Transport Layer Security,* или *TLS*) уже не раз упоминался на страницах этой книги, начиная с TLS и DTLS для MQTTи CoAP и заканчивая сетевой безопасностью в WAN и PAN. Каждый из этих протоколов так или иначе полагается на TLS. Стандарт TLS вобрал в себя все криптографические протоколы и технологии, которые мы уже обсуждали. В этом разделе мы кратко пройдемся по спецификации TLS1.2, ее структуре и использованию.

Изначально уровень защищенных сокетов (англ. *Secure Sockets Layer,* или *SSL*) был представлен в 1990 г., но уже через 9 лет ему на смену пришла технология TLS. С 2008 г. текущая спецификация TLS1.2 входит в состав стандарта RFC5246. TLS 1.2 включает в себя генератор хешей SHA-256, который был добавлен вместо SHA-1 для улучшения безопасности.

Процесс шифрования в TLS выглядит следующим образом:

1) клиент открывает соединение с сервером, поддерживающим TLS (порт 443 для HTTPS);
2) клиент предоставляет список шифров, которые он поддерживает;
3) сервис выбирает шифр и функцию шифрования и оповещает клиент;
4) сервер передает клиенту цифровой сертификат, выданный удостоверяющим центром и содержащий открытый ключ;
5) клиент подтверждает подлинность сертификата;
6) для генерации ключа сеанса используется один из двух способов:
   1) серверу передается случайное число, предварительно зашифрованное с помощью его открытого ключа. Затем сервер и клиент создают на его основе ключ сеанса, который используется на протяжении взаимодействия;
   2) ключ сеанса для шифрования и дешифрования генерируется с помощью протокола Диффи-Хеллмана. Полученный ключ используется, пока не закроется соединение.
7) взаимодействие переходит в зашифрованный канал.

На рис. 12.9 показан процесс рукопожатия двух устройств, взаимодействующих через TLS1.2.

Протокол датаграмм безопасности транспортного уровня (англ. *Datagram Transport Layer Security,* или *DTLS*) – это коммуникационный протокол на основе TLS, который работает поверх UDP (версия DTLS 1.2 основана на TLS 1.2). Он предназначен для обеспечения похожих гарантий безопасности и используется в легковесном протоколе CoAP.

Программно-определяемый периметр

Ранее, в главе 8, мы обсуждали такие понятия, как программно-определяемые оверлейные сети. Способность оверлейной сети создавать микросегменты является чрезвычайно полезной, особенно при массовом масштабировании IoT-устройств и в ситуациях, когда есть возможность нивелировать последствия DDoS-атаки. Программно-определяемые сети имеют дополнительный компонент под названием *SDP* (*Software-Defined Perimeter* – программно-определяемый периметр), который заслуживает отдельного внимания в контексте безопасности оверлейных сетей.

Архитектура программно-определяемого периметра

Программно-определяемый периметр (англ. *Software-Defined Perimeter,* или *SDP*) – это подход к построению сети, который не предусматривает никакой модели доверия. Он основан на так называемом черном облаке *Департамента систем защиты информации* (англ. *Defense Information Systems Agency,* или *DISA*). В черном облаке обмен информацией происходит лишь по мере необходимости. SDP может смягчить последствия таких атак, как DDoS, MITM, эксплойты нулевого дня, сканирование серверов и т. д. Помимо предоставления оверлея и микросегментации для каждого подключенного устройства, периметр

ограждает пользователей, клиенты и IoT-устройства специальным барьером, проникнуть через который можно только по приглашению (на основе проверки подлинности).

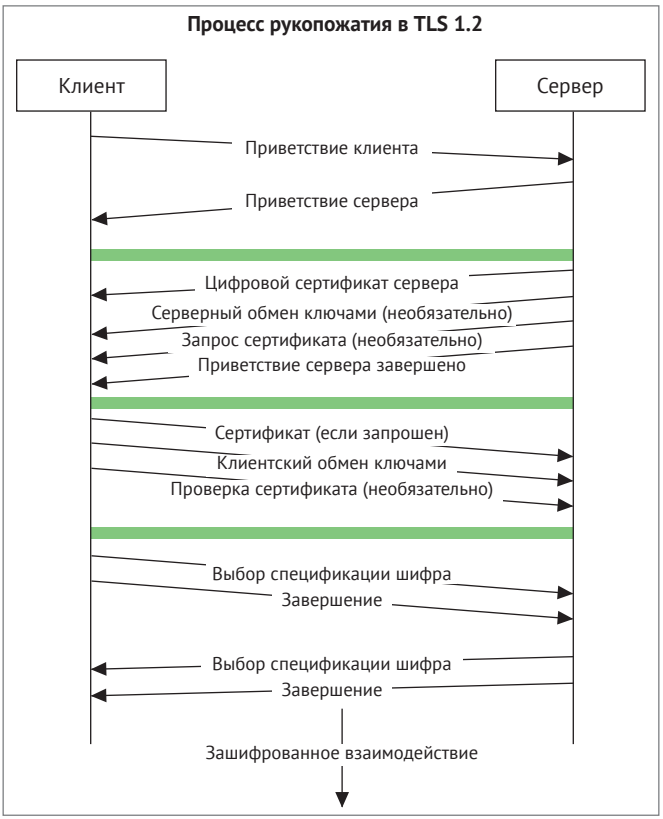


Рис. 12.9 ❖ Последовательность шагов при рукопожатии в TLS 1.2

SDP можно использовать для создания оверлейной сети (это сеть, построенная поверх другой сети). Например, когда-то интернет-сервисы были основаны на существующих телефонных коммуникациях. В такой гибридной модели распределенный управляющий уровень остается неизменным. Пограничные маршрутизаторы и виртуальные коммутаторы направляют данные в зависимости от правил, описанных на управляющем уровне. Поскольку устойчивость сети SDN во многом аналогична проводным сетям, она идеально подходит для приложений реального времени, удаленного мониторинга и сложной обработки событий. Возможность создания нескольких оверлейных сетей поверх одних и тех же пограничных компонентов позволяет выполнять *микросегментацию*, открывая потребителям прямой доступ к различным ресурсам.

Каждая пара «ресурс–потребитель» представляет собой неизменяемую сеть; ее доступ за пределы виртуального оверлея определяется системным администратором.

Возможность создания нескольких оверлейных сетей поверх одних и тех же пограничных компонентов позволяет выполнять микросегментацию, когда каждая конечная точка в глобальной распределенной сети IoT может формировать отдельные изолированные сегменты, используя имеющуюся инфраструктуру. Теоретически мы можем изолировать каждый датчик. Это мощный инструмент, который позволяет подключаться к IoT-устройствам как сервисам, используя соединения промышленного уровня, а также изолировать и защищать их друг от друга.

На рис. 12.10 показан пример SDN-оверлея. У некой корпорации есть три удаленных магазина с рядом разных IoT-устройств и пограничных компонентов в каждом. Сеть основана на SDN-оверлее с микросегментами, изолирующими системы POS и VOIP; те, в свою очередь, корпоративно управляются с помощью датчиков, предназначенных для мониторинга безопасности, страховых условий и состояния холодильной камеры. Сторонние поставщики услуг могут управлять различными удаленными датчиками, каждый из которых находится в изолированной и безопасной виртуальной оверлейной сети.

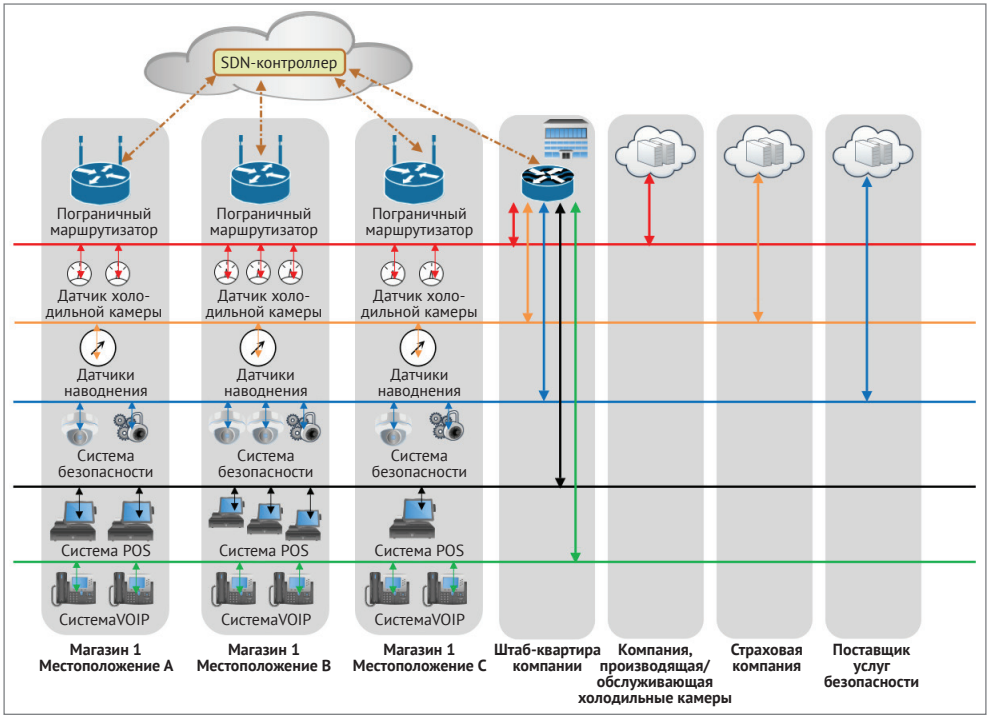


Рис. 12.10 ❖ Пример оверлейной SDN-сети

Чтобы усилить безопасность SDP, можно разработать систему приглашений, заставляя парные устройства аутентифицироваться перед подключением. В сеть могут быть добавлены только предварительно авторизованные пользователи и клиенты. Управляющий уровень может расширить этот подход, предоставляя приглашения через электронную почту или какой-то механизм регистрации. Если пользователь примет приглашения, клиентские сертификаты и полномочия будут распространяться лишь на ту систему, с которой он работает. Система приглашений ведет список расширенных сертификатов и предоставляет оверлейное соединение только в случае, если обе стороны приняли предложенные им роли.

В качестве аналогии из реального мира можно привести рассылку приглашений на вечеринку. Приглашения отправляются отдельным друзьям по почте, с указанием даты, времени, адреса и прочих подробностей. Каждый из адресатов сам решает, принимать ли приглашение. Как вариант, вечеринку можно прорекламировать по интернету, телевизору или радио; в этом случае каждый посетитель проверяется на входе.

Блокчейн и криптовалюта в интернете вещей

Блокчейн (англ. *blockchain*) – это публичный, цифровой, децентрализованный реестр (англ. *ledger*) или цепочка криптовалютных транзакций. Первым криптовалютным блокчейном является Bitcoin, но помимо него на рынке существует более 700 новых валют, таких как Ethereum, Ripple и Dash. Сильная сторона этой технологии заключается в отсутствии единой сущности, контролирующей состояние транзакций. Она также обеспечивает избыточность системы, заставляя каждого участника хранить копию реестра. Если предположить, что участники системы не склонны доверять друг другу, их взаимодействие должно быть основано на консенсусе.

Напрашивается вопрос: зачем передавать данные или валюту в блокчейне, если мы уже решили проблемы проверки подлинности и безопасности с помощью ассиметричной криптографии и обмена ключами? Дело в том, что передача денежных средств и ценной информации требует чего-то большего. Представьте, что у нас есть два устройства (назовем их Боб и Элис). Согласно теории информации, когда Боб передает Элис какое-то сообщение или фрагмент данных, он сохраняет у себя копию переданной информации. При обмене деньгами или контрактами данные должны покинуть источник и появиться в пункте назначения. Они должны существовать в единственном экземпляре. Проверка подлинности и шифрование обеспечивают взаимодействие, но нам необходим новый инструмент для передачи владения (рис. 12.11).

Безопасные криптовалюты на основе блокчейна имеют большое значение для интернета вещей. Можно привести такие примеры:

- **прямые денежные платежи между устройствами** – интернет вещей должен быть готов к поддержке устройств, обменивающих услуги на валюту;

- **управление цепочками снабжения** – неизменяемость и безопасность блокчейна может пригодиться в логистике, инвентаризации и перемещении товара, делая бумажный учет ненужным. Все контейнеры, перемещения, местоположения и состояния могут отслеживаться, проверяться и сертифицироваться. Попытки подмены, удаления или изменения учетной информации становятся невозможными;
- **солнечная энергия** – представьте солнечную энергию в виде услуги. В этом случае на крыше жилого дома устанавливаются солнечные панели, которые могут не только генерировать электроэнергию для жильцов, но и поставлять ее в общую электросеть (например, в обмен на так называемые углеродные кредиты).

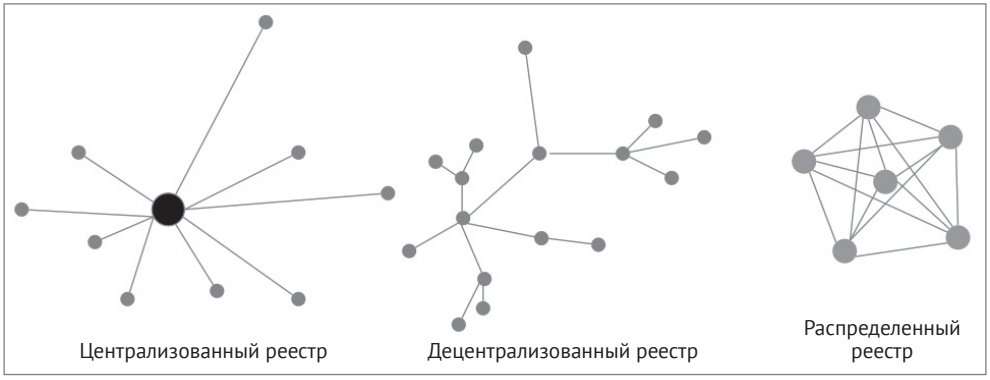


Рис. 12.11 ❖ Топология реестра. Централизованный реестр – это типичный подход, в котором обслуживанием «журнала» занимается единая сущность. В криптовалютах используются либо децентрализованные, либо распределенные реестры

Bitcoin (блокчейн)

Та часть Bitcoin, которая относится к криптовалюте, не является блокчейном как таковым. Bitcoin – это искусственная валюта, которая сама по себе не имеет никакой ценности и ничем не обеспечена (в отличие от золота). Ее нельзя пощупать; она существует лишь в рамках сети. И, наконец, количество «монет» Bitcoin не контролируется центральным банком или правительством. Это полностью децентрализованная технология. Как и другие блокчейны, она основана на криптографии открытого ключа, большой и распределенной одноранговой сети и протоколе, который определяет структуру Bitcoin. В 2008 г. Сатоши Накамото (псевдоним) опубликовал в рассылке, посвященной криптографии, документ под названием «Биткойн: система цифровой пиринговой наличности». В 2009 г. была запущена первая сеть Bitcoin, в которой Сатоши сгенерировал первый блок (первичный блок [англ. *Genesis Block*]).

Концепция блокчейна подразумевает наличие *блока*, который представляет собой текущий фрагмент реестра. Компьютер, подключенный к сети блокчейн,

называется *узлом* (англ. *node*). Каждый узел участвует в проверке подлинности и передаче транзакций; для этого он получает копию реестра и, в сущности, становится ее администратором.

Распределенные сети, основанные на одноранговых топологиях, идеально подходят для Bitcoin. В сети Bitcoin действует закон Меткалфа, поскольку ее размер определяет ценность валюты. Сеть хранит цепочку записей (реестр). Возникает вопрос: кто захочет добровольно делиться своими вычислительными ресурсами для мониторинга журнала? Ответом является система вознаграждения на основе *майнинга* (от англ. *mining*).

На рис. 12.12 показан процесс выполнения транзакции. Вначале делается запрос, который транслируется по пиринговой (англ. *Peer-to-Peer,* или *P2P*) сети компьютеров (*узлов*). Эта сеть ответственна за проверку подлинности своих пользователей, во время которой также проверяется сама транзакция. Затем транзакции объединяются в новый блок данных в распределенном реестре. После заполнения блок добавляется в существующий блокчейн и больше не подлежит изменению. Ниже проиллюстрированы аутентификация, майнинг и проверка подлинности в Bitcoin.

На диаграмме показан процесс передачи 0,000554 BTC между Элис и Бобом со служебной комиссией 0,0001 BTC. Элис инициирует транзакцию, подписывая ее содержимое комбинацией хеша предыдущей транзакции и своего закрытого ключа. Элис также помещает свой закрытый ключ в скрипт inputScriptSig. Затем транзакция распространяется по сети для дальнейшего добавления в блок и проверки подлинности. Каждый участник сети пытается первым подтвердить подлинность и найти подходящее случайное число (NONCE), основанное на текущем уровне сложности. При нахождении блока сервер передает его другим участникам, чтобы те его проверили и добавили в цепочку.

Ниже представлен качественный анализ блокчейна в целом и работы Bitcoin в частности. Важно понимать эти фундаментальные принципы, основанные на всех тех средствах безопасности, которые мы рассмотрели ранее в этой главе:

- **транзакция с цифровой подписью** – Элис хочет передать Бобу 1 Bitcoin. Для начала об этом нужно публично объявить. Для этого Элис пишет сообщение «Элис отдаст Бобу 1 Bitcoin» и подтверждает его цифровой подписью на основе своего закрытого ключа. Любой обладатель открытого ключа может проверить подлинность данного сообщения. Однако Элис может повторить свое сообщение и тем самым подделать денежные средства;
- **уникальная идентификация** – чтобы решить проблему с подлогом, Bitcoin создает уникальный серийный номер, как это делает Американское казначейство на своих банкнотах. Для этого вместо числа, которое назначается централизованным способом, используется хеш. Этот хеш автоматически генерируется во время транзакции и позволяет ее идентифицировать.

  Еще одна серьезная проблема связана с двойной тратой. Даже если транзакция подписана и имеет уникальный хеш, Элис может попытаться пере-

дать тот же Bitcoin другим участникам. Боб проверит транзакцию, инициированную Элис, и у него все сойдется. Но, если Элис выполнит ту же транзакцию, только по отношению к Чарли, она, в сущности, обведет систему вокруг пальца. Сеть Bitcoin очень большая, но возможность хищения средств в ней хоть и незначительна, но все же присутствует. Чтобы защититься от двойной траты, пользователи Bitcoin, принимающие платежи через блокчейн, ждут подтверждения. Со временем появляется все больше подтверждений, что повышает шанс успешного прохождения проверки;

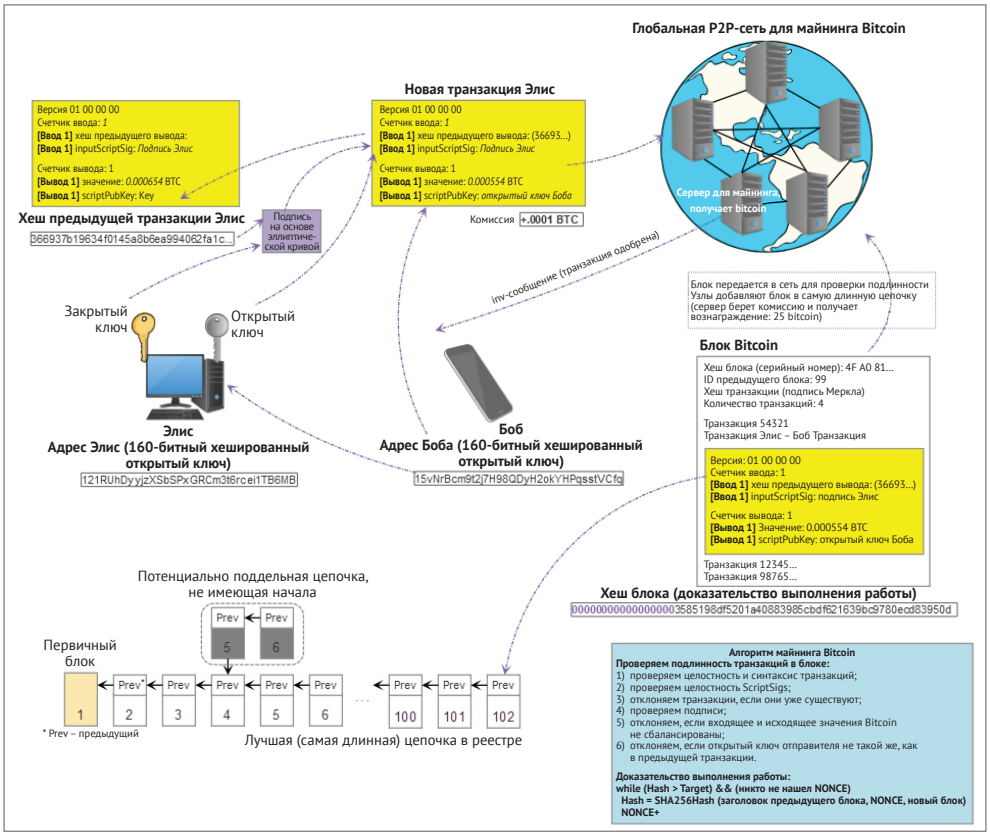


Рис. 12.12 ❖ Процесс выполнения транзакций в блокчейне Bitcoin

- **проверка подлинности другими узлами** – чтобы исключить двойную трату в блокчейне, получатели транзакции (Боб и Чарли) передают информацию о платеже в сеть и просят других участников проверить ее подлинность. Такая проверка выполняется не безвозмездно;
- **доказательство выполнения работы** – проблема двойной траты все еще не решена до конца. Элис может заполучить контроль над сетью

с помощью собственных серверов и заявить, что все ее транзакции являются подлинными. Чтобы исключить такую возможность раз и навсегда, в Bitcoin была добавлена концепция *доказательства выполнения работы* (англ. *Proof of Work,* или *PoW*). Она имеет два аспекта. Во-первых, проверка аутентичности транзакции должна требовать значительных вычислительных ресурсов. Это должно быть нечто более сложное, чем сопоставление ключей, имен пользователей, идентификаторов транзакций и других тривиальных атрибутов аутентификации. Во-вторых, пользователи должны поощряться за помощь в подтверждении денежных транзакций других участников (см. следующий шаг);

- чтобы заставить пользователей, проверяющих транзакции, выполнять определенную работу, к заголовкам транзакций добавляется случайно сгенерированное число. Bitcoin хеширует это число вместе с сообщением в заголовке, используя безопасный алгоритм SHA-256. Этот хеш называется *целевым*; он имеет размер меньше 256 бит, а его содержимое постоянно меняется. Чем меньше это значение, тем больше ресурсов тратится на поиск оригинального сообщения. Поскольку каждый хеш, в сущности, генерирует абсолютно случайное число, пользователям приходится вычислять множество значений типа SHA-256. В среднем на каждое значение уходит примерно 10 минут.

  Доказательство выполнения работы такой продолжительности означает, что проверка транзакции в среднем будет занимать те же 10 минут. Майнеры работают с блоками, состоящими из множества транзакций. На текущий момент размер блока не превышает 1 Мб, и пока он не будет обработан, ваша транзакция не сможет завершиться. Это может иметь последствия для IoT-устройств, которые должны работать в режиме реального времени.

- **стимулы для майнинга Bitcoin** – чтобы поощрить построение пиринговой сети для проверки чужих транзакций, пользователи вознаграждаются за проделанную работу. Есть делается двумя способами. Первый – через майнинг Bitcoin, от которого выигрывают участники, занимающиеся проверкой блоков с транзакциями. Второй способ заключается в выплате комиссии за проведенную транзакцию. Комиссию получает майнер, который помогает проверить подлинность блока. Изначально комиссия была равна нулю, но с ростом популярности Bitcoin она тоже начала расти. В среднем успешная транзакция вознаграждается 35 долларами (в виде BTC). Если обработка блока происходит в ускоренном режиме, комиссию можно поднять. Таким образом, даже после вычисления всех хешей в текущем поколении Bitcoin, у пользователей будет стимул поддерживать транзакции;

  Изначально вознаграждение было очень высоким (50 BTC), но после вычисления первых 210 000 блоков оно уменьшается вдвое каждых четыре года. Так будет продолжаться до 2140 г., пока частота уменьшения не достигнет критической

точки; после этого объем вознаграждения будет меньше самой мелкой единицы измерения Bitcoin(10^{-8} BTC), которую еще называют сатоши (*satoshi*).
Учитывая, что каждых 10 минут вычисляется новый блок, а вознаграждения снижается вдвое каждые четыре года, в какой-то момент мы можем достичь максимального количества возможных блоков. К тому же, мы знаем, что изначально награда составляла 50 BTC. Из этого можно вывести последовательность, которая стремится к лимиту сатоши: 50 BTC + 25 BTC + 12,5 BTC… = 100 BTC. Итого, максимальное количество «монет» Bitcoin равно 210 000 * 100 = 21.

- **безопасность на основе порядка сцепления блоков** – порядок, в котором выполняются транзакции, тоже имеет огромное значение для целостности Bitcoin. Если средства передаются от Элис к Бобу, а затем от Боба к Чарли, эти события должны быть записаны в реестре именно в таком порядке. Для этого транзакции в блокчейне *сцепляются*. Каждый новый блок, попадающий в сеть, содержит указатель на последний блок в цепочке, который был проверен. В Bitcoin транзакция не является действительной, пока ее не добавят в самую длинную цепочку, и пока вслед за ней не будет подтверждено хотя бы пять других блоков. Это решает проблему с асинхронностью, если Элис попытается передать Бобу и Чарли одни и те же средства. Она может попробовать сообщить о каждой транзакции отдельным наборам майнеров. Однако в момент схождения сети этот обман будет выявлен. Таким образом, если транзакция успешно завершится для Боба, сеть объявит ее недействительной в случае с Чарли. Но правила порядка сцепления остановят Элис, даже если вместо Чарли она попытается заплатить самой себе. Допустим, она передаст Бобу 1 BTC и дождется подтверждения транзакции (пять блоков спустя), а затем переведет те же средства на свой адрес, вызвав тем самым раздвоение цепочки. После этого ей придется подтвердить пять дополнительных блоков. Согласно пункту 4, это займет около 50 минут и потребует огромных вычислительных ресурсов, так как она должна будет опередить всех других майнеров вместе взятых

Еще одним интересным аспектом блокчейна является метод борьбы с DDoS-атаками, который основан на доказательстве выполнения работы (система «протокол/функция») и является экономической мерой. Злоумышленник пытается насытить сеть как можно большим объемом данных и тем самым вывести систему из строя. Блокчейн с механизмом PoW снижает эффективность такой атаки. Ключевой момент этого подхода состоит в его асимметричности: работа, проделываемая запрашивающей стороной, должна быть достаточно трудоемкой (но выполнимой), и при этом поставщик услуг должен иметь возможность быстро ее проверить.

IOTA (направленный ациклический граф)

Недавно появилась новая интересная криптовалюта под названием IOTA, предназначенная специально для интернета вещей. Ее архитектура основана на *направленном ациклическом графе* (англ. *Directed Acyclic Graph,* или *DAG*), а цепочка

доверия формируется самими IoT-устройствами. Bitcoin предусматривает комиссию за каждую транзакцию. В IOTA нет никаких комиссий. Это делает возможным проведение микротранзакций, что очень важно в контексте интернета вещей. Например, множество клиентов могут подписаться на показания датчика через MQTT. В целом, эта услуга имеет определенную ценность, но каждая отдельная транзакция является настолько незначительной, что комиссия, снимаемая в сети Bitcoin, была бы выше, чем стоимость самих данных.

Архитектура IOTA имеет следующие особенности:

- контроль над средствами не централизован; в блокчейне пользователи могут объединяться в большие группы, чтобы увеличить количество блоков, которые они могут сгенерировать, и соответствующее вознаграждение. Это может привести к концентрации влияния и навредить сети;
- нет необходимости в дорогом оборудовании. Для майнинга криптовалюты в сети Bitcoin нужны мощные процессоры, способные справиться со сложными вычислениями;
- микро- и нанотранзакции на уровне отдельных IoT-устройств;
- надежная защита от взлома методом простого перебора, даже если при этом используются квантовые компьютеры;
- через IOTA можно передавать не только валюту, но и данные. При этом имеется полноценная поддержка аутентификации и защита от подмены;
- в сети IOTA содержимое транзакции может быть каким угодно, поэтому на ее основе можно построить национальную систему голосования, защищенную от стороннего вмешательства;
- роль сервиса может выполнять любое устройство с компактным чипом. IOTA позволяет арендовать что угодно: дрель, персональный маршрутизатор, микроволновку или велосипед – для этого требуется лишь наличие небольшого чипа или микроконтроллера.

Граф DAG в IOTA называется *клубком* (англ. *tangle*) и используется для хранения транзакций в виде распределенного реестра. Транзакции используются узлами (IoT-устройствами) и составляют клубок DAG. Если транзакции A и B не соединены напрямую направленной гранью, но из A в B можно провести маршрут длиной не меньше расстояния между этими двумя точками, считается, что A косвенно подтверждает B.

Существует также понятие *первичной* транзакции. Так как клубок не может быть начат путем майнинга граней графа и при этом отсутствуют стимулы и комиссии, каждый узел должен содержать все токены; первичное событие рассылает их по адресам «основателей». Это статическое множество всех токенов, которое никогда больше не будет пополняться.

Каждая новая транзакция должна подтвердить (или отклонить) предыдущие две; этот процесс формирует в графе прямую грань и называется прямым подтверждением. Для произведения транзакции необходимо выполнить «работу» от имени клубка. Работа заключается в поиске случайного числа (nonce), которое подходит для хеша фрагмента подтвержденной транзакции. Таким

образом, благодаря использованию IOTA сеть становится более распределенной и безопасной. Транзакции могут быть подтверждены много раз. С увеличением их количества растет уверенность в их правомочности. При попытке одобрить неправомочную транзакцию, узел рискует отклонением собственной транзакции и исключением из клубка.

Эта технология все еще находится на ранней стадии развития, но за ней стоит понаблюдать. Больше информации можно найти на сайте iota.org.

Правовое регулирование

Государственные регулирующие органы рекомендуют и устанавливают стандарты, без соблюдения которых производители не могут называть свою продукцию безопасной. С ростом количества атак на IoT-системы этой темой заинтересовалось правительство США; теперь устройства, подключенные к сети, должны поддерживать определенные стандарты безопасности. Архитекторы, занимающиеся глобальным масштабированием IoT-устройств, должны ориентироваться в этих правилах и законах, так как похожие нормы могут быть введены и в других странах. Эти законы касаются конфиденциальности и безопасности на уровне физических лиц и общества в целом.

Законопроект об улучшении безопасности интернета вещей (август, 2017)

Безопасностью интернета вещей заинтересовалось несколько правительств. 1 августа 2017 г. в Американском Сенате был представлен кросспартийный законопроект (S.1691 – Закон об улучшении безопасности интернета вещей, www.congress.gov/bill/115th-congress/senate-bill/1691/text). Это попытка формализовать и отрегулировать минимальные стандарты безопасности для устройств, подключаемых к интернету, которые покупаются правительством США. И, хотя этот законопроект еще не принят, он четко показывает, как именно собираются регулировать IoT-устройства.

В частности, текст документа предъявляет следующие требования к подрядчикам, поставляющим правительству решения на основе интернета вещей:

- прошивка, а также аппаратное и программное обеспечение не должны иметь уязвимостей, перечисленных в базе данных института NIST;
- программное обеспечение и прошивка должны поддерживать аутентифицированные обновления и заплатки. Кроме того, подрядчики обязаны своевременно исправлять уязвимости. Подрядчик отвечает за поддержку развернутых систем с указанием периода и условий обслуживания IoT-устройств;
- для взаимодействия, шифрования и взаимной связи нельзя использовать устаревшие протоколы и технологии;
- в код нельзя встраивать учетные данные для удаленного администрирования;

- устройство, подключенное к интернету, должно уметь обновлять или исправлять любой компонент своего программного обеспечения или прошивки в случае уязвимости;
- подрядчик должен предоставить письменный сертификат, удостоверяющий, что любые сторонние технологии, поставляемые вместе с IoT-устройством, соблюдают данные стандарты;
- глава Административно-бюджетного управления может установить крайние сроки для удаления или замены имеющихся IoT-устройств, которые признаны небезопасными;
- в течение 60 дней с момента принятия закона глава Национального управления защиты и программ при содействии частных и академических технологических специалистов должен опубликовать формальные методические рекомендации по кибербезопасности для всех без исключения IoT-устройств, которые используются федеральным правительством.

Правительственным организациям позволено внедрять или продолжать использовать технологии, уровень безопасности которых превышает тот, что указан в законопроекте (после утверждения главой Административно-бюджетного управления). Кроме того, законопроект учитывает тот факт, что IoT-устройства имеют жесткие ограничения относительно вычислительной мощности и памяти и могут не соответствовать заявленным стандартам. В таких случаях главе управления можно направить прошение о послаблении требований с планом дальнейшего обновления и замены. Законопроект позволяет вводить в эксплуатацию следующие технологии (для минимизации риска необходимо совместное одобрение со стороны института NIST и главы управления):

- программно-определяемая (микро)сегментация сети;
- контейнеры и микросервисы для изолированных сред выполнения, управляемые на системном уровне;
- многофакторная аутентификация;
- «разумные» пограничные сетевые решения, такие как шлюзы с поддержкой изоляции и устранения угроз.

И, хотя на момент написания этих строк законопроект все еще находится на рассмотрении комитета, он, безусловно, свидетельствует о том, что федеральные власти обеспокоены безопасностью и уязвимостями IoT-устройств. Законопроект может быть изменен или даже отклонен, но проблема защиты интернета вещей уже находится в поле зрения Американского правительства и законодателей.

Другие правительственные учреждения

Другие департаменты правительства США уже опубликовали нормы и рекомендации для целого ряда технологий вокруг интернета вещей. Больше других отличился *Национальный институт стандартов и технологий* (англ. *National Institute for Standards and Technology,* или *NIST*), который составил несколько документов и руководств по защите устройств, подключенных к сети. Он

также занимаются поддержкой национальных и международных стандартов безопасности. Вспомогательные материалы можно найти на сайте csrc.nist.gov. Ниже перечислено несколько важных документов, относящихся к криптографии и стандартам FIPS:

- NIST Special Publication 800-121 Revision 2, «Guide to Bluetooth Security». Определяет рекомендуемые нормы безопасности для протоколов Bluetooth Classic и Bluetooth BLE: nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-121r2.pdf;
- NIST Special Publication 800-175A, «Guidelines for Using Cryptographic Standards in the Federal Government». Рекомендации по применению криптографических стандартов в федеральном правительстве: nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-175A.pdf;
- NIST FIPS standards. Стандарты FIPS: csrc.nist.gov/publications/search?requestSeriesList=3&requestStatusList=1,3&requestDisplayOption=brief&requestSortOrder=5&itemsPerPage=All.

Министерство внутренней безопасности США (англ. *Department of Homeland Security,* или *DHS*) предоставляет обязательные правила эксплуатации для всех федеральных структур, связанных с национальной безопасностью, в том числе и в области информационных технологий. Недавно была опубликована директива 18-01, которая предписывает «кибергигиену» в сфере работы с электронной почтой, управления ключами, *идентификации сообщений, создания отчетов и определения соответствия по доменному имени* (англ. *Domain-based Message Authentication, Reporting, and Conformance,* или *DMARC*), веб-безопасности с применением исключительно HTTPS и т. д. DHS также участвует в создании предписаний относительно стандартов кибербезопасности для Конгресса США, других министерств и частного сектора.

Компьютерная группа реагирования на чрезвычайные ситуации (англ. *Computer Emergency Response Team,* или *US-CERT*) тоже играет важную роль в сфере безопасности. С 2000 г. US-CERT отвечает за поиск, изолирование, публикацию и остановку угроз кибербезопасности на национальном уровне. Ее сотрудники занимаются цифровой судебной экспертизой, подготовкой персонала, мониторингом в режиме реального времени, созданием отчетов и действенной защиты от уязвимостей нулевого дня и актуальных угроз безопасности. Текущие сигналы тревоги и описание предпринимаемых мер можно найти здесь: www.us-cert.gov/ncas/alerts.

Рекомендации по защите IoT-устройств

В интернете вещей безопасность должна учитываться с самого начала, а не задним числом, после завершения проектирования или введения в эксплуатацию – на этих этапах будет уже слишком поздно. Кроме того, подход к безопасности должен быть комплексным и покрывать все аспекты: от аппаратного обеспечения до облака. В этом разделе рассматривается простой проект IoT с защитой, которая пронизывает все его уровни, начиная с датчика и закан-

чивая облачной инфраструктурой. Мы попытаемся развернуть его с разными мерами предосторожности, чтобы усложнить задачу потенциальным злоумышленникам.

Комплексная безопасность

Если сосредоточиться на каком-то одном аспекте интернета вещей, итоговая цепочка безопасности будет иметь слабые звенья. Безопасность должна пронизывать все уровни системы: от датчика и до облака. Это комплексный подход. Каждый компонент в цепочке управления и данных должен иметь контрольный список параметров безопасности и потенциальных угроз. На рис. 12.13 показан пример такой многоуровневой системы защиты, которую следует предусмотреть при развертывании.

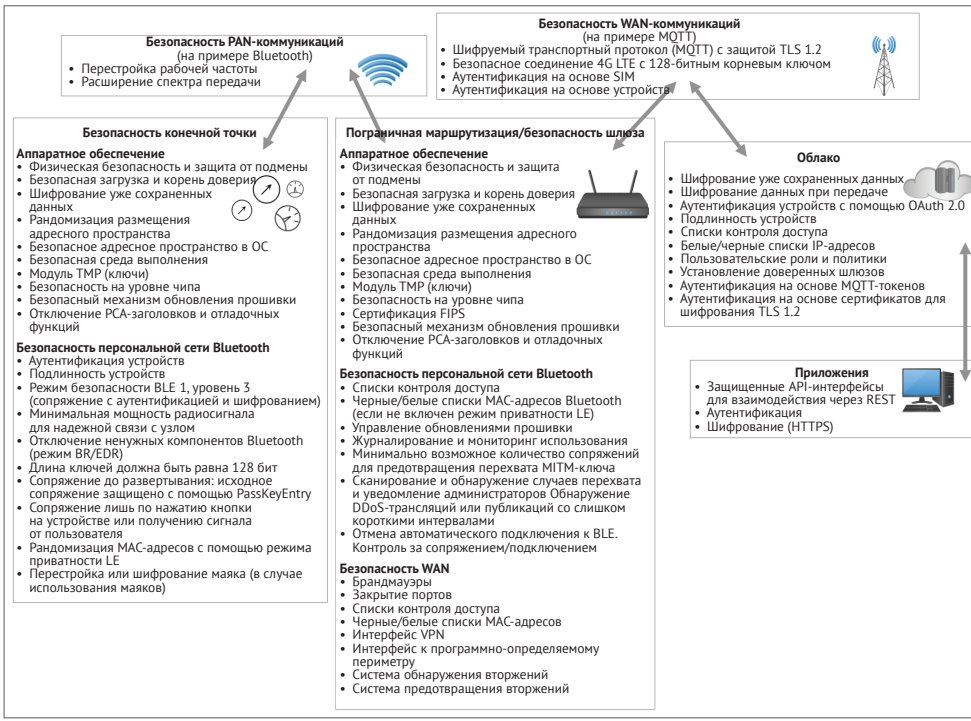


Рис. 12.13 ❖ Комплексная защита, начиная с датчика и заканчивая облаком. Здесь приводится пример датчиков с поддержкой Bluetooth, которые, в основном, взаимодействуют с облачным сервисом через пограничный шлюз. Целостность и безопасность должны обеспечиваться на каждом уровне. Это относится как к аппаратным, так и к программным компонентам. Физические устройства должны быть защищены от подмены, а радиосигналы – от глушения и DDoS-атак; оборудование для корня доверия и ASLR предотвращают внедрение кода, данные шифруются, при сопряжении и связывании используется аутентификация, сеть проложена через VPN-серверы и брандмауэры и т. д.

Краткий перечень мер безопасности

Ниже приведен список проверенных временем рекомендаций и идей, относящихся к безопасности. Также важно иметь комплексную защиту:

- используйте самые последние версии операционных систем и библиотек со всеми необходимыми заплатками;
- используйте аппаратное обеспечение с поддержкой таких функций защиты, как безопасные среды выполнения, модули TPM и неисполняемые адресные пространства;
- обфускация кода в надежде, что злоумышленник не сможет его распутать, – относительно безнадежная затея. Подписывайте, шифруйте и защищайте свои прошивки и программные образы – особенно те, которые доступны на вебсайте компании;
- выбирайте исходный пароль случайным образом;
- используйте корень доверия и безопасную загрузку, чтобы гарантировать, что на устройствах ваших клиентов работает подлинное программное обеспечение;
- уберите пароли из кода прошивок;
- все IP-порты должны быть закрыты по умолчанию;
- используйте рандомизацию размещения адресного пространства, стековые индикаторы и безопасные сегменты памяти, которые поддерживаются в современных ОС;
- используйте автоматические обновления. Предоставьте производителям механизм для исправления ошибок и уязвимостей в рабочих системах. Для этого программная архитектура должна быть модульной;
- планируйте вывод из эксплуатации заранее. IoT-устройства могут работать долго и продуктивно, но когда-то их придется утилизировать. Это включает в себя безопасное удаление и уничтожение всех модулей постоянной памяти (flash) в устройстве;
- предлагайте своим клиентам и пользователям вознаграждение за найденные ошибки – особенно те, которые могут привести к появлению уязвимостей нулевого дня;
- подпишитесь на уведомления об активных угрозах, которые рассылает US-CERT, чтобы всегда быть в курсе об актуальных эксплойтах и кибератаках;
- построение проекта на основе таких простых (и небезопасных) протоколов, как MQTT или HTTP, может показаться заманчивым, но вы должны поставлять свои устройства с включенной аутентификацией на основе TLS или DTLS. Шифруйте данные, начиная с датчика и заканчивая облаком;
- используйте антиотладочные фьюз-биты. Детонируйте их на этапе производства для безопасной отладки, прежде чем выпускать продукт.

Заключение

В этой главе мы подробно рассмотрели риски безопасности в интернете вещей. Имея в виду наличие таких известных вирусов как Mirai и Stuxnet, специально нацеленных на IoT-устройства, архитекторы IoT-систем должны с самого начала заботиться о безопасности своих архитектур. Интернет вещей является идеальной средой для выполнения разного рода атак. Обычно системы этого типа обладают менее зрелой защитой по сравнению с ПК. IoT-устройства представляют собой наиболее обширную поверхность атаки на планете, а удаленность некоторых из них позволяет злоумышленникам получить физический доступ к оборудованию, немыслимый в безопасных офисных условиях. Эти угрозы требуют серьезного внимания, так как их последствия могут затронуть как отдельные устройства, так и города или даже целые страны.

Дополнительный материал на эту тему можно найти по следующим ссылкам:

- Black hat: www.blackhat.com;
- Defcon: www.defcon.org;
- Digital Attack Map: www.digitalattackmap.com;
- Gattack: gattack.io;
- IDA Pro Interactive Disassembler: www.hex-rays.com;
- RSA Conference: www.rsaconference.com;
- Shodan: www.shodan.io.

В конце этой книги приводится список консорциумов и организаций, которые принимают участие в развитии интернета вещей и сопутствующих технологий, правовых норм и стандартов.

Глава 13

Консорциумы и сообщества

Промышленные консорциумы необходимы для популяризации, регулирования и разработки стандартов. Сфера интернета вещей в этом отношении не отличается от других технологий, для неё также разрабатываются отраслевые и открытые стандарты. В этой главе мы расскажем о различных консорциумах, касающихся **персональных вычислительных сетей** (**PAN**), протоколов, глобальных вычислительных сетей, туманных и граничных вычислений, а также о различных специализированных консорциумах (umbrella). Мы подробно и всесторонне расскажем о каждом альянсе, чтобы IoT-архитектор мог решить, к какому из них стоит присоединиться и стоит ли присоединяться вообще, – ведь это требует временных затрат и денежных вложений. Необходимо отметить, что организация не обязана состоять в каком-либо промышленном объединении, множество великолепных продуктов и компаний были построены без опоры на консорциумы. Однако некоторым организациям все же нужно членство в профессиональном объединении для того, чтобы получить возможность пользоваться логотипом или даже самим выпускать определенные стандарты.

Интернет вещей – это развивающийся сегмент рынка, вызывающий большой общественный интерес, поэтому на раннем этапе его развития вокруг него будут образовываться разнообразные альянсы, которые станут инструментом борьбы за право разработки стандартов. Это естественное явление на стадии интенсивного развития любой отрасли. Часто альянсы формируются в тот момент, когда один стандарт соревнуется с другим похожим стандартом, и организации делятся на противоборствующие лагери. Еще бывает так, что отраслевые стандарты определяются некоммерческими и научными учреждениями. Список консорциумов из этой главы как минимум подскажет архитектору, где искать источники и технические материалы, необходимые для выполнения проекта.

В этой главе рассказывается об особенностях, истории и членстве в различных организациях и консорциумах в IoT-пространстве, включая информацию о системе передачи данных, стандартах облачных и туманных вычислений.

Консорциумы по персональным сетям

Персональные вычислительные сети (как основанные на IP-протоколе, так и без его участия) собрали вокруг себя несколько консорциумов и отраслевых

комитетов. Во главе многих из них стоят партнеры-основатели, и, чтобы пользоваться их наработками, необходимо в них вступать или получать разрешение.

Bluetooth SIG

Описание компании:

- **сайт** – www.bluetooth.org;
- **год основания** – 1998;
- **количество членов** – 20 000.

Консорциум Bluetooth SIG образовался в 1998 г., его основали четыре компании-соучредителя: Ericsson, Nokia, Toshiba и IBM. К завершению 1998 г. в организации числилось уже 400 членов. Миссия организации – продвигать стандарты, развивать рынки, организовывать форумы и распространять информацию о технологиях Bluetooth. Организация стремится к дальнейшему развитию, а также к лицензированию и к регистрации товарных знаков Bluetooth. С точки зрения организационной структуры консорциум SIG разделен на несколько небольших целевых групп: исследовательские группы для научной деятельности, экспертные группы, специализирующиеся на нескольких сферах приложения технологии Bluetooth, рабочие группы, которые занимаются разработкой новых стандартов, и комитеты, в задачу которых входит лицензирование и маркетинг. Членство бывает нескольких типов: промоутеры, которые влияют на общее направление развития и занимают руководящие должности в SIG; младшие члены, которые могут присоединиться к рабочей группе, получать первые и продвинутые варианты спецификаций, знакомиться с маркетинговыми материалами, посещать мероприятие PlugFest и получать квалификационные списки; третий тип – пользовательское членство, единственный бесплатный вариант, такие члены не могут присоединяться к рабочим группам.

Thread Group

Описание компании:

- **сайт** – www.threadgroup.org;
- **год основания** – 2014;
- **количество членов** – 182.

Консорциум Thread Group изначально был основан корпорациями Alphabet (холдинг, которому принадлежит компания Google), Samsung, ARM, Qualcomm, NXP и ещё шестью. Thread – это протокол персональных сетей, основанный на стандарте 6LoWPAN, базой которого является стандарт 802.15.4. Рабочая группа лицензирует протокол Thread с помощью модели лицензирования для свободного программного обеспечения BSD. Задача группы – составить прямую конкуренцию протоколу Zigbee, особенно в области персональных сетей на технологии MESH. В консорциуме три уровня корпоративного членства. Нижний уровень членства, партнерский, позволяет организации пользоваться логотипом, общаться с прессой и получать документацию. Члены среднего уровня, участники, могут присоединяться к рабочим группам и комитетам,

а также участвовать в тестовых испытаниях. Членство высшего уровня, спонсорское, подразумевает место в правлении и участие в распределение бюджета организации.

Альянс Zigbee

Описание компании:

- **сайт** – www.zigbee.org;
- **год основания** – 2002;
- **количество членов** – 446.

Альянс Zigbee, занимающийся вопросами, связанными с протоколом Zigbee, который появился в 1998 г. Он определяет правила работы самоорганизующихся и самовосстанавливающихся сетей с mesh-топологией. Сети Zigbee, как и Thread, работают поверх уровня 802.15.4, но не на основе IP-протокола. Программный стек по-прежнему основан на лицензии GPL, ранее поступало несколько запросов сделать процесс лицензирования более гибким. Консорциум предлагает три уровня членства. Соразработчики одними из первых получают доступ к спецификациям, посещают конференции и могут пользоваться логотипом Zigbee. Участники могут вступать и работать в различных технических комитетах, а также высказывать свое мнение по поводу спецификаций. И, наконец, промоутеры могут входить в правление альянса, право ратифицировать спецификации есть только у них.

Другое

Среди других подобных организаций можно назвать:

- **Альянс DASH7** – отвечает за развитие протокола DASH7 (www.dash7-alliance.org);
- **ModBus** – промышленные консорциум, развивающий протокол Modbus, широко используемый в промышленности (www.modbus.org);
- **Альянс Z-Wave** – отраслевое и административное объединение, занимающееся технологиями, построенными на протоколе Z-Wave (z-wave-alliance.org).

Консорциумы по протоколам

Эти организации занимаются протоколами верхнего уровня, такими как MQTT. Хотя многие протоколы являются открытыми, например, MQTT, членство в консорциумах позволяет принимать участие в разработке и голосовании по вопросам новых стандартов.

Open Connectivity Foundation и Allseen Alliance

Описание компании:

- **сайт** – www.openconnectivity.org;
- **количество членов** – 300;
- **год основания** – 2014.

Организация Open Connectivity Foundation изначально называлась Open Interconnect Foundation, но в 2016 г., после выхода из состава участников компании Samsung и присоединения новых членов, была переименована в OCF. На протяжении двух лет это была отдельная организация, но в 2016 г. произошло слияние OCF и объединения Allseen Alliance. Их общая миссия заключается в создании платформ межоперационного взаимодействия для потребителей, компаний и целых сфер деятельности путем разработки стандартов, фреймворков и сертификационных программ под брендом Open Connectivity Foundation. Эта работа охватывает множество сегментов: автомобили, бытовая электроника, корпорации, здравоохранение, домашняя автоматика, промышленности и портативные устройства. Наибольшую популярность фреймворкам этой организации принес набор спецификаций Universal Plug and Play (UPnP), а также проект IoTivity и протокол взаимодействия AllJoyn. Все они используют модель лицензирования Internet Systems Consortium (ISC), которая с функциональной точки зрения задумывалась как эквивалент BSD. Существует пять уровней членства с различными ежегодными взносами. Базовое членство бесплатно для всех, оно дает доступ к тестовым инструментам и предоставляет возможность читать спецификации. Членство для некоммерческих образовательных организаций предоставляет доступ к рабочим группам и процессу сертификации. Кроме того, есть членство золотого, платинового и бриллиантового уровней, каждый из которых отличается от других степенью доступа к ресурсам OCF: от участия в рабочих группах до членства в правлении.

OASIS

Описание компании:

- **сайт** – www.oasis-open.org;
- **количество членов** – 300;
- **год основания** – 1993.

OASIS (англ. *Organization for the Advancement of Structured Information Standards*) – большая некоммерческая организация, которая была образована в 1993 г. Она участвовала в разработке десятков стандартных языков и протоколов и отвечает за решения, касающиеся протоколов MQTT и AMQP, которые широко применяются в IoT-сообществе. Технологии этого консорциума затрагивают интернет вещей, облачные вычисления, энергетический сектор и аварийное управление,– и это только несколько изнаправлений. OASIS предлагает три типа членства. Членство на уровне участника обеспечивает безграничное участие во всей деятельности комитета. На спонсорском уровне появляются преимущества визуального и маркетингового порядка, такие как демонстрация межоперационной совместимости и возможность использовать логотип. И, наконец, уровень спонсора-учредителя предлагает самую высокую степень прозрачности и такие бонусы, как презентации продуктов OASIS для сотрудников компании и денежные поощрения – стипендии и гранты. Лидеры отрасли, развивающие открытые стандарты, обычно вступают в консорциум в качестве

спонсора-учредителя. Размер ежегодного взноса зависит от типа членства и количества сотрудников.

Object Management Group

Описание компании:

- **сайт** – www.omg.org;
- **количество членов** – 250;
- **год основания** – 1989.

Консорциум Object Management Group (OMG) изначально создавался как некоммерческая организация, в состав которой вошли корпорации Hewlett Packard, IBM, Sun, Apple, American Airlines и Data General. Задачей этого партнерства была работа над различными объектно-ориентированными стандартами в сфере вычислительных технологий. Самые известные стандарты, разработанные этим консорциумом, – это UML и CORBA, а недавно на арене интернета вещей консорциум OMG занял руководящую роль в Консорциуме промышленного интернета. OMG принимает участие в различных направлениях развития промышленного интернета вещей и программно-определяемых сетей. К целевым аспектам интернета вещей относятся распределенные информационные системы, который обеспечивают интероперабельность, а также управление угрозами в сфере интернета вещей. Модель управления консорциумом представляет собой структуру из трех сегментов: архитектурный совет, комитет по платформенным технологиям и комитет по доменам. Предлагается пять уровней членства. Членство аналитического уровня предоставляется только научно-исследовательским компаниям, которые получают доступ к ряду собраний по вопросам доменов и платформенных технологий. Пробное членство – это наиболее бюджетный вариант для всех остальных компаний, он также подразумевает приглашения на собрания некоторых комитетов. Следующие два уровня – это платформенный и доменный, которые предоставляют компаниям право голоса в соответствующих комитетах. Последний вариант – это вступить в консорциум на правах участника, получив доступ ко всем комитетам, право занимать место в правлении и пр.

IPSO Alliance

Описание компании:

- **сайт** – www.ipso-alliance.org;
- **количество членов** – 60;
- **год основания**: 2008.

Эта группа не занимается разработкой стандартов. Ее задачей является популяризация протокола IP как средства подключения умных объектов к интернету и как средства решения проблем межоперационной совместимости. Объединение оказывает помощь и поддержку Инженерному совету интернета через различные рабочие группы, занимающиеся семантикой (стандарты метаданных для различных объектов), IoT-протоколами и анализом различных стандартов, безопасности и приватности.

Существует три уровня членства: новаторский, который предназначен для компаний, состоящих из 10 сотрудников и меньше; уровень участника, который предоставляет компании доступ к техническим проектам и комитетам; и промоутерский уровень, который предоставляет право занимать место в правлении и другие бонусы.

Другое

Среди других организаций, деятельность которых направлена на IoT-протоколы и безопасность, можно назвать:

- **Online Trust Alliance**: OTA– это всемирная некоммерческая организация, разрабатывающая лучшие решения для IoT-безопасности (otalliance.org);
- **oneM2M**: общие сервисы, протоколы и архитектура подключенных объектов (www.onem2m.org).

Консорциумы по глобальным вычислительным сетям

Эти организации занимаются различными протоколами и технологиями передачи данных на дальние расстояния (LPWAN). На некоторые из этих технологий нужно приобретать право пользования, а другие бесплатны.

Weightless SIG

Описание компании:

- **сайт** – www.weightless.org**;**
- **количество членов** – неизвестно;
- **год основания** – 2012.

Некоммерческая организация Weightless SIG была создана для финансовой и технической поддержки различных протоколов Weightless LPWAN. В сферу интересов SIG входят три стандарта: Weightless-N (для бюджетных и энергосберегающих систем), Weightless-P (для полноценной двусторонней связи) и Weightless-W, который предлагает широкий диапазон возможностей и функций. Цель – разработка стандартов для таких ситуаций, как отслеживание транспортных средств, умные счетчики и даже широкополосная сеть в сельской местности. Область интересов консорциума Weightless преимущественно составляют медицина и промышленность. Хотя стандарты Weightless являются открытыми, необходимо пройти через квалификационную процедуру. Члены консорциума могут бесплатно пользоваться протоколами Weightless IP и получают доступ к программе сертификации.

Вариант вступления в консорциум всего один: в качестве разработчика.

LoRa Alliance

Описание компании:

- **сайт** – www.lora-alliance.org**;**
- **количество членов** – 419 (44 дополнительных формальных члена);
- **год основания** – 2014.

LoRa Alliance – это некоммерческая организация, которая занимается развитием технологии LPWAN: LoRaWAN. LoRaWAN – это архитектура протокольного уровня для дальней связи при спектре частот до одного ГГц. LoRaWAN обычно применяется для межмашинного взаимодействия и в системах «умный город». Существует четыре платных уровня членства для компаний – уровень соразработчика, который позволяет пользоваться сертифицированными продуктами ассоциации, предоставляет доступ к итоговым документам и дает возможность участвовать в ряде собраний. Следующий уровень – институциональное членство, которое позволяет состоять в рабочих группах и получать черновые документы по проектам. Уровень участника наделяет правом голосовать в рабочих группах. И, наконец, спонсорский уровень, который является основанием для получения места в правлении ассоциации, предоставляет доступ к штатным данным и возможность возглавлять рабочую группу.

Инженерный совет интернета

Описание компании:

- **сайт** – www.ietf.org;
- **количество членов** – 1200 (общее количество присутствующих на собраниях IETF);
- **год основания** – 1986.

Изначально состоявший из 21 исследователя из США, Инженерный совет интернета (англ. *Internet Engineering Task Force*, *IETF*) существенно разросся, и к настоящему времени он превратился в организацию, контролирующую все события, касающиеся таких стандартов, как TCP/IP, и различных документов RFC. Теперь сфера деятельности Инженерного совета очень широка и затрагивает IoT-сегмент, например, протоколы LPWAN, 6lo и IPVS на базе стандарта 802.15.4. В состав Инженерного совета интернета входит Группа по выработке инженерного регламента интернета (IESG), которая руководит процессом стандартизации. С точки зрения структуры, он делится на семь направлений, таких как маршрутизация или транспортировка. Каждое направление, в свою очередь, может делиться на десятки различных рабочих групп (в общей сложности, в совете насчитывается более 140 рабочих групп). Чтобы вступить в Инженерный совет интернета, нужно просто подписаться на рассылку рабочей группы и проявлять активность, разбираться в деятельности совета и стандартах, а также участвовать в жизни совета. Стандартизация – это очень скрупулезная процедура, поскольку эта ассоциация устанавливает базовые принципы связи через интернет.

Wi-Fi Alliance

Описание компании:

- **сайт** – www.wi-fi.org;
- **количество членов** – 700;
- **год основания** – 1999.

Wi-Fi Alliance – это некоммерческая организация, занимающаяся разработкой стандартов. Она была основана в середине 1990-х гг., и в ее задачу входило решение вопросов, касающихся интероперабельности при беспроводной связи. Появление стандарта 802.11b привело к учреждению промышленного консорциума, который стал эффективным руководящим органом. Альянс регулирует процесс сертификации беспроводных технологий, и устройства, отвечающие стандартам ассоциации, получают право на соответствующий логотип и товарный знак. В альянсе 19 направлений работы и специализированных групп, включая такие направления, как 802.11ax, безопасность и интернет вещей.

В ассоциации три уровня членства. Уровень разработчика, который позволяет компании применять прошедшие сертификацию продукты в своих решениях Партнерский уровень присваивается организациям, находящимся под контролем спонсора или участника ассоциации, и наделяет эту организацию всеми правами члена ассоциации. Третий уровень членства – вступить в ассоциацию в качестве участника, что предполагает возможность вносить свой вклад в процесс сертификации и определение требований новых технологий.

Консорциумы по туманным и граничным вычислениям

Туманные и граничные вычисления применяются все шире, в связи с чем был разработан двойной набор соответствующих стандартов. Для решения растущего числа проблем межоперационной совместимости в области туманных вычислений необходимы отраслевые стандарты и система отраслевой организации. В данном разделе рассказывается о нескольких ассоциациях, занимающихся разработкой стандартов и фреймворков, направленных на улаживание вопросов интероперабельности.

OpenFog

Описание компании:

- **сайт** – www.openfogconsortium.org;
- **количество членов** – 55;
- **год основания** – 2015.

OpenFog – это некоммерческое общественно-государственное объединение, которое занимается разработкой стандартов и внедрением туманных вычислений. Группа направляет деятельность других организаций по стандартизации в таких сферах, как интероперабельность в облачных технологиях, безопасность и совместная разработка стандартов. Консорциум OpenFog состоит в партнерстве с ассоциациями OPC и Институтом инженеров электротехники и электроники (IEEE), а руководит организацией совет из 15 членов.

В состав консорциума входит технический комитет, который отвечает за работу шести рабочих групп, специализирующихся на архитектуре, тестовых испытаниях и других аспектах процесса разработки. Также в консорциуме есть комитет по маркетингу, занимающийся внешними контактами и привлече-

нием внимания. Консорциум предлагает бесплатное членство для научно-образовательных учреждений и членство для правительственных/специальных организаций. Обычный корпоративный тип членства, позволяющий оказывать влияние и принимать участие в подготовке стандартов, предусматривает возможность вступления в консорциум и для небольших компаний с доходом менее 50 миллионов долларов США.

EdgeX Foundry

Описание компании:

- **сайт** – www.edgexfoundry.org;
- **количество членов** – 50;
- **год основания** – 2017.

Консорциум EdgeX Foundry отвечает за обеспечение работы модульной платформы, предназначенной для обеспечения взаимодействия между устройствами и операционной системой в рамках IoT-системы посредством открытого программного обеспечения. В число микросервисов входит межплатформенное ПО для процессоров правил, оповещения, регистрации данных, ведения учета и подключения устройств. Проект курируется консорциумом Linux Foundation, а его продукты распространяются по лицензии Apache. Компания Dell отвечает за структуру исходного кода, который представляет собой серию микросервисов, подходящих для любого устройства. Предусмотрено три уровня членства: младший, серебряный и платиновый. Младшим членом может стать только некоммерческая организация, серебряные члены могут выбирать по одному представителю в правление компании, а платиновые члены имеют право на представителя в правлении и нескольких комитетах. Все члены могут участвовать в общих собраниях и других событиях.

Специализированные организации

Организации, о которых пойдет речь, руководят или оказывают влияние на множество технических и функциональных аспектов интернета вещей (а также других сегментов). Они занимаются различными вопросами, связанными с протоколами, тестированием, эксплуатационной пригодностью, технологией, обменом данными и теорией.

Консорциум промышленного интернета

Описание компании:

- **сайт**: www.iiconsortium.org;
- **количество членов** – 258;
- **год основания** – 2014.

Некоммерческая организация, которая была основана в 2014 г. компаниями AT&T, Cisco, GE, IBM и Intel. Консорциум призван объединять промышленных партнеров и способствовать внедрению и развитию промышленного интерне-

та вещей. Группа не занимается стандартизацией, но устанавливает эталоны архитектуры и тестовых испытаний в сфере производства, медицины, перевозок, умных городов и энергетики. На данный момент консорциум включает в себя 19 рабочих групп, в область задач которых входят обеспечение связи, безопасность, энергетика, умные фабрики и медицина. В консорциуме шесть уровней членства, включая правительственный и некоммерческий уровни. Корпоративное членство предполагает разную степень влияния и стоимость в зависимости от ежегодного оборота компании. Организация выпускает очень подробную документацию по тестовым испытаниям, которая содержит конкретные примеры применения, например, тестирование системы обслуживания багажа в авиакомпании. Как уже говорилось ранее, ассоциация OMG контролирует деятельность Консорциума промышленного интернета, но последний является отдельной организацией.

Институт инженеров по электротехнике и электронике IoT (IEEE IoT)

Описание компании:

- **сайт** – iot.ieee.org;
- **количество членов** – нет данных;
- **год основания** – 2014.

IEEE IoT – это не консорциум, а группа по специальным интересам под эгидой Института инженеров по электротехнике и электронике (IEEE). Это многопрофильная организация, включающая в себя учебные учреждения, органы управления и специалистов в области промышленности и инженерии, в задачу которых входит развитие интернета вещей. Организация IEEE IoT влияет или разрабатывает самостоятельно специализированные IoT-стандарты, такие как набор стандартов для протоколов 802.15 и набор стандартов для беспроводных сетей 802.11. Организация проводит бесплатные вебинары, занятия и готовит интернет-материалы, пополняющие информационную базу о данной сфере. Группа организует всемирные конференции, практические курсы и совещания, а также выпускает один из наиболее популярных научных журналов: IEEE Internet of Things Journal.

Другое

Среди других специализированных организаций можно назвать:

- **Genivi** – группа, занимающаяся открытым ПО для бортовой информационно-развлекательной системы и автомобилей с сетевыми возможностями (www.genivi.org);
- **HomeKit** – стандарты для устройств Apple, стандарты управления умным домом с помощью мобильных устройств (developer.apple.com/homekit/);
- **HomePlug** – организация, занимающаяся техническими стандартами для приложений, обслуживающих систему умного дома и пользовательских приложений (www.homeplug.org/);

- **Open Automotive Alliance** – группа, в сферу интересов которой входят автомобили и технологии. Группа занимается вопросами применения операционной системы Android в транспортных средствах (www.openautoalliance.net/#about);
- **Wireless-Life Sciences Alliance** – беспроводные системы связи в медицине и промышленности (wirelesslifesciences.org/).

Американские правительственные организации по вопросам IoT и безопасности

Ниже перечислены правительственные и федеральные организации, которые должны быть вам знакомы, особенно если вы занимаетесь безопасностью в сфере интернета вещей:

- **Национальный институт стандартов и технологий** – занимается определением государственных стандартов безопасности, шифрования и связи (www.nist.gov);
- **Министерство внутренней безопасности** (Консультативный комитет по связи в системе национальной безопасности, NSTAC) – занимается укреплением кибербезопасности и инфраструктуры всемирной связи (www.dhs.gov/national-security-telecommunicationsadvisory-committee);
- **Национальное управление по телекоммуникациям и информации** – представляет собой одно из подразделений Министерства торговли США и занимается такими вопросами, как распределение частот в США, назначение доменных имен и безопасность (www.ntia.doc.gov/home);
- **Компьютерная команда экстренной готовности США** (US-CERT) – выявляет и реагирует на угрозы компьютерной безопасности государственного масштаба (www.us-cert.gov/ncas/current-activity).

Заключение

Консорциумы и промышленные объединения приносят сообществу ощутимую пользу – разрабатывают стандарты, технические регламенты и добиваются интероперабельности. Вступление в консорциум дает организации неограниченный доступ к спецификациям и документации. Во многих случаях стать членом или партнером консорциума – это обязательное условие для получения права пользования результатами его деятельности. Со стратегической точки зрения, чем больше в консорциуме участников, тем сильнее его влияние, поскольку различные протоколы и стандарты конкурируют с друг другом за свое место в IoT-пространстве.

Предметный указатель

Л

М

Н

О

П

Т

У

Книги издательства «ДМК Пресс» можно заказать
в торгово-издательском холдинге «Планета Альянс» наложенным платежом,
выслав открытку или письмо по почтовому адресу:
115487, г. Москва, 2-й Нагатинский пр-д, д. 6А.
При оформлении заказа следует указать адрес (полностью),
по которому должны быть высланы книги;
фамилию, имя и отчество получателя.
Желательно также указать свой телефон и электронный адрес.
Эти книги вы можете заказать и в интернет-магазине: **www.a-planeta.ru**.
Оптовые закупки: тел. **(499) 782-38-89**.
Электронный адрес: **books@alians-kniga.ru**.

Перри Ли

Архитектура интернета вещей

Главный редактор *Мовчан Д. А.*
dmkpress@gmail.com
Перевод *Райтман М. А.*
Корректор *Светлова О. Ю.*
Верстка *Чаннова А. А.*
Дизайн обложки *Мовчан А. Г.*

Формат 70×100 1/16.
Гарнитура «PT Serif». Печать офсетная.
Усл. печ. л. 37,05. Тираж 200 экз.

Веб-сайт издательства: **www.dmkpress.com**