Глава 5

Будущее начинается сегодня

> Мы должны признать, что успешная архитектура имеет свойство развиваться, поскольку успех ведет к росту, а технология движется вперед. Вследствие этого различные допущения должны периодически пересматриваться при обновлении протоколов.
>
> *RFC 6250[1], май 2011 г.*

В предыдущих главах мы говорили об эволюции Интернета как платформы, где инновации происходят на различных уровнях в различные временные периоды. Мы также говорили об оссификации, или окаменелости, некоторых уровней и приложений — в первую очередь на уровне IP.

Сегодня мы наблюдаем наиболее бурное развитие именно на уровне приложений и связанного с ними контента. И хотя сетевой и транспортный уровни Интернета выполняют основную функцию обеспечения глобальной связности, новые приложения требуют от коммуникационной среды параметров, которые эти уровни напрямую не в состоянии предоставить.

Однако это не является проблемой, а скорее показывает работоспособность модели Интернета — низшие уровни не перегружаются дополнительной функциональностью, которая может быть реализована на более высоких уровнях.

В этой главе мы рассмотрим два примера. Они иллюстрируют, как на базе фундаментальной инфраструктуры могут быть построены целые экосистемы с параметрами,

[1] RFC 6250: Evolution of the IP Model, URL: https://www.rfc-editor.org/rfc/rfc6250

которых немыслимо ожидать от «простого» Интернета. В качестве первого примера возьмем сети доставки контента — CDN. Здесь мы посмотрим, как параметры высокой масштабируемости, надежности и качества могут быть достигнуты путем создания оверлейных систем. Второй пример расскажет об Интернете вещей, или IoT (Internet of Things). В случае IoT мы являемся свидетелями эволюции, когда обыденные вещи начинают видеть, думать и действовать. Мы наблюдаем внедрение проникающих систем управления и контроля — в индустрии, офисе и дома. Это расширяет наши возможности, но таит и существенные риски.

О новых технологиях, протоколах и системах, которые позволяют нам лучше представить, что ждет нас завтра, — в этой главе.

Что такое сети доставки контента и зачем они нужны?

Как мы обсуждали в предыдущих главах, основной задачей опорной инфраструктуры Интернета является пересылка пакетов от одного узла к другому в надежде, что пакет достигнет своего места назначения в целости и в срок. Но Сеть сама никаких гарантий на этот счет не дает. Если пакет окажется утерянным вследствие перегрузки какого-либо участка Сети или будет доставлен с опозданием, вне изначального порядка (например, вследствие недоступности основного маршрута), — то приложениям придется обрабатывать такие нерегулярности соответствующим образом. Пропускная полоса коммуникационного канала между приложениями также не гарантирована и заранее не известна. Этот параметр определяется множеством факторов, начиная с пропускной способности отдельных сегментов сетей, через которые передается трафик между приложениями, и заканчивая загрузкой самих сетей.

В начале 90-х годов прошлого столетия большинство приложений Интернета были весьма «эластичными», а именно малочувствительными к параметрам передачи. Например, обмен файлами с помощью протокола FTP или отображение веб-страницы могли занять секунды, минуты или часы, но фундаментального значения для приложения это не имело. Новые же приложения — приложения реального времени, интерактивные приложения и просто современные веб-порталы — потребовали определенных стабильных параметров качества.

В главе 3 мы говорили о попытках разработать и внедрить решения по «повышению качества» в Интернете. Эти решения были стандартизованы, однако их внедрение потерпело полное фиаско. Во многом потому, что они требовали удорожания и усложнения опорной инфраструктуры, пересмотра существующих взаимоотношений по обмену трафиком между операторами, но также и потому, что в конце 1990-х значительно упала стоимость пользования каналами передачи данных, в том числе и международными. Соответственно, возросла и скорость доступа для конечного пользователя.

Тем не менее случаи, когда для одного пользователя портал онлайн-магазина загружается за две секунды, а для другого — за несколько минут, являлись вполне типичными. Еще пример: после минуты нормального воспроизведения видеоролик внезапно замирает, пусть и на секунды. Такие ситуации, конечно, нельзя назвать приемлемыми.

Но Интернет — это сложная система взаимодействующих сетей, которые управляются независимо и различаются топологически и технологически. Возьмите хотя бы такой простой фактор, как расстояние. В таблице 1 показано влияние расстояния между клиентом (веб-браузером) и сервером на пропускную способность виртуального канала стандартного TCP.

Таблица 7. Влияние расстояния на пропускную способность

Масштаб расстояния	Время RTT	Типичная потеря пакетов	Эффективная пропускная способность	Время скачивания 4ГБ DVD
Местный (<160 км)	1,6 мс	0,6%	44 Мб/с (высококачественный поток HDTV)	12 мин
Региональный (900-1800 км)	16 мс	0,7%	4 Мб/с (минимальный поток HDTV)	2,2 часа
Межконтинентальный 5000 км	48 мс	1,0%	1 Мб/с (поток телевидения стандартного качества)	8,2 часа
Многоконтинентальный 10 000 км	96 мс	1,4%	0,4 Мб/с (видеопоток слабого качества)	20 часов

Источник: Erik Nygren, Ramesh Sitaraman and Jennifer Sun
The Akamai Network: A Platform for High-Performance Internet Applications
ACM SIGOPS Operating Systems Review Vol. 44 Iss. 3 (2010)

Перефразируя известную поговорку, если пользователь не может забрать необходимый контент, контент должен быть доставлен пользователю! Так появилась идея сетей доставки контента — CDN (Content Delivery Networks) — надстройки или еще одного приложения Интернета.

Краткая история CDN

Появлению первых CDN в конце 1990-х предшествовало внедрение таких технологий, как серверные фермы с балансировкой загрузки, иерархические кеши и веб-прокси. Они позволяли улучшить производительность веб-сервера и доставку контента. В Массачусетском технологическом институте (MIT) Том Лейтон (Tom Leighton) и Данни Левин (Danny Lewin) разработали математические алгоритмы для оптимальной маршрутизации и репликации контента в широкомасштабной сети распределенных серверов. Эти алгоритмы впоследствии легли в основу архитектуры крупнейшей CDN Akamai.

Внедрению и развитию первых CDN способствовало новое явление, получившее название «внезапной толпы» (Flash crowd), также называемое эффектом Slashdot. Суть его сводится к тому, что на малозаметный доселе веб-сайт вдруг обрушивается шквал популярности, который делает его практически недоступным вследствие перегрузки как самого сайта, так и сетевой инфраструктуры. Так, многие новостные сайты не выдержали нагрузки после известных событий 9/11 в США.

Первое название, Flash crowd, произошло от одноименного фантастического романа Ларри Нивена (Larry Niven), написанного в 1973 году. По его сюжету, была изобретена телепортация, и люди смогли свободно и мгновенно переноситься в те места, где происходит что-нибудь любопытное. Второе название происходит от новостного сайта Slashdot[2], публикация дайджеста в котором со ссылкой на оригинальный сайт иногда давала эффект, схожий с атакой отказа в обслуживании из-за наплыва посетителей.

В 1999 году Akamai запустила коммерческую службу доставки контента, клиентом которой стал один из наиболее посещаемых сайтов — Yahoo!

По мере роста потребности в качественной доставке различных видов контента другие компании также занялись построением CDN. В качестве наиболее известных примеров можно привести Limelight Networks (2001), EdgeCast (2006), Amazon (2008), CloudFlare (2009), Incapsula (2009).

Первое поколение CDN было направлено на улучшение производительности веб-сайтов и обслуживало преимущественно статический контент. Основным подходом к построению таких сетей являлось размещение кеширующих серверов (серверных ферм) в максимальной близости от провайдеров доступа, обслуживающих конечных пользователей. Наиболее типичным являлось обеспечение присутствия в точках обмена трафиком.

С появлением услуг видео по требованию, обеспечивающих потоковую передачу, встала необходимость в архитектурном подходе, отличном от традиционных CDN. Дело в том, что видеоклип по существу мало отличается от большого файла, но его передача должна быть синхронизирована с потоковым сервером, воспроизведение может быть начато и прервано в произвольный момент.

Архитектура CDN

Идея CDN проста: разместить реплики контента как можно ближе к его потенциальным потребителям и тем самым обеспечить стабильную необходимую пропускную способность, а также распределить нагрузку на серверы-поставщики контента, например, на веб-серверы. Реализация этой идеи в глобальном Интернете гораздо сложнее и требует ответа на вопросы: где и сколько реплик должно быть размещено, как синхронизировать контент, каким образом осуществлять перенаправление запросов пользователя к нужной реплике?

[2] https://slashdot.org

Решение связанных с этим задач требует создания распределенной наложенной сети с функциями управления и мониторинга. Можно выделить следующие основные компоненты CDN, представленные на рис. 79.

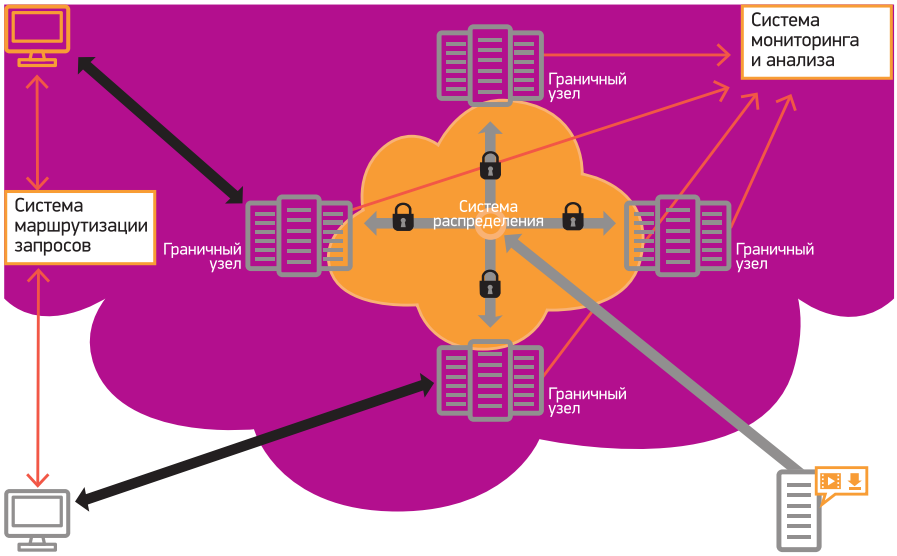

Рис. 79. Основные компоненты CDN.

Граничные кеширующие узлы

Кеширующие узлы, объединенные в наложенную или выделенную сеть, обеспечивают доставку контента на «последней миле». От того, насколько географически распространена сеть и насколько «правильно» расположены граничные узлы, зависит общая производительность и качество CDN.

Граничные узлы могут представлять собой компьютерные кластеры для обеспечения надежности и распределения нагрузки. Обычно они размещаются вблизи точек обмена трафиком, где можно обеспечить связность CDN со множеством сетевых операторов.

Многие крупные CDN, такие как Akamai, Google, Netflix и Facebook, используют так называемые внутрисетевые (on-net) кеши внутри сетей доступа, которые взаимодействуют с пограничными серверами и доставляют контент еще ближе к конечному пользователю. С распространением сетей 5G и облачных услуг на основе граничных вычислений с множественным доступом (Multi-access edge computing, MEC) такие сетевые кеши могут располагаться еще ближе к пользователю — на границе базовой сети оператора мобильной связи. Распределение контента в эти локальные кеши является многоуровневым, что еще существеннее минимизирует дублирование трафика и задержки.

Запросы клиентов обслуживаются самими узлами локально, если запрашиваемые данные находятся в кеше. При их отсутствии — данные копируются в кеш от исходного сервера. Однако кеширующие узлы отличаются от стандартного прокси, поскольку могут выполнять ряд других функций, перечисленных ниже.

1. **Управление кешем**
   Разные классы контента требуют различных характеристик кеша. На основе статистической популярности и метаданных контента (например, заголовков Last-Modified) узел может вычислить возможное время истечения годности кеша и заранее освежить данные от источника. Это особенно важно в случае, когда исходный сервер не предоставляет информацию по управлению кешем (с помощью заголовков cache-control, etag, expires). Но даже если сервер использует заголовки для управления кешем, они рассчитаны на кеширование браузером и не всегда адекватно работают в условиях CDN.

   Поскольку CDN располагает значительной информацией относительно конкретного контента — его типа, характера запросов и т.п., управление кешем может обеспечиваться более динамично, нежели с помощью заданных статических параметров кеша на сервере. Например, даже динамический контент, который обычно не подлежит кешированию, имеет определенный период годности, в течение которого граничный узел может отвечать на запросы клиентов без обращения к исходному серверу.

2. **Определение местонахождения источника контента, включая обработку ситуаций недоступности**
   С помощью системы маршрутизации запросов, о которой мы поговорим далее, узел может принять решение о выборе оптимального экземпляра исходного сервера, модификации запроса (модификации URL) или перенаправлении запроса на другой граничный узел.

3. **Изменение HTTP-заголовков**
   Граничные узлы могут добавлять, удалять или изменять заголовки HTTP-запросов и ответов, особенно заголовки куки (cookie), такие как set-cookie и cookie. Это может использоваться для обмена служебной информацией с сервером-источником и с пользователями, а также для управления каскадными кешами.

Транспортная система доставки контента

В задачу транспортной системы входит оптимальная доставка требуемого контента к граничным кеширующим узлам. Небольшие CDN могут себе позволить одноуровневую структуру, когда граничные узлы посылают запросы на обновление кеша непосредственно веб-серверам. При этом для связности часто используется Интернет, а не выделенная сеть.

Крупные CDN используют многоуровневую структуру, когда группы граничных узлов приписаны к кеширующим кластерам более высокого уровня. Таким образом уменьшается нагрузка на веб-серверы — источники контента за счет увеличения частоты попадания в кеш (cache hit), а также уменьшается задержка доставки за счет дополнительного уровня распределения.

В качестве транспортной сети в крупных CDN используется либо наложенная, либо выделенная сеть. Во многих случаях применяется оптимизация маршрутизации с использованием непрерывных тестов связности и пропускной способности между узлами. Все чаще для управления передачей трафика в транспортной сети используется архитектура SDN, о которой мы говорили в главе 3 «Глобальная система маршрутизации и передачи данных».

Более сложная структура иерархических кешей используется при необходимости доставки видеопотоков, особенно в реальном времени. От граничных узлов также требуется дополнительная функциональность, так как они играют роль потоковых серверов. Потоковые серверы обеспечивают необходимую фрагментацию потока, его кодирование с различным разрешением/качеством и взаимодействие с клиентами.

Например, в сети Akamai в качестве узлов транспортной сети используется система «входных узлов» (entrypoint), расположенных в непосредственной близости к поставщику контента, и рефлекторов (set reflector), обеспечивающих оптимальную доставку данных в граничные узлы.

Система маршрутизации запросов

«Мозгом» CDN является система маршрутизации запросов. В задачи этой системы входит выбор оптимального граничного узла для обслуживания запроса клиента. Выбор осуществляется на основе нескольких параметров, включая близость узла к клиенту, загрузку узла, наличие в кеше требуемого контента, а также других возможных правил (например, связанных с защитой авторских прав или доступности услуг в определенных географических областях). Для достижения этой цели используется либо DNS, либо Anycast BGP. Далее мы рассмотрим оба метода более подробно.

Работа CDN

При отсутствии CDN, когда пользователь пытается получить доступ к онлайн-контенту, например, к веб-серверу, он направляется на исходный сервер. В простой настройке имя хоста в URL-адресе, например, https://www.example.com, преобразуется DNS в IP-адрес исходного сервера (93.184.216.34). Как только соединение HTTP(S) установлено, содержимое доставляется пользователю.

Когда веб-сервер использует CDN для доставки контента, пользователь должен быть направлен на оптимальный граничный узел, а не на исходный сервер. Эту задачу решает система маршрутизации запросов.

Когда пользователь запрашивает контент с граничного узла, а контент там уже существует, он будет предоставлен немедленно. Это называется «кеш-попадание» (cache hit). Если запрошенный контент просрочен в кеше пограничного сервера или никогда не кешировался, граничный узел должен будет сначала загрузить его с исходного сервера. Это называется «промах кеша» (cache miss). Хотя это может привести к некоторому снижению производительности, обычно контент, связанный с запрашиваемым, также подгружается, поэтому дальнейшие запросы с большой вероятностью будут попадать в кеш.

Маршрутизация запросов с использованием DNS

Этот метод является наиболее популярным вследствие повсеместного использования DNS. В его основе лежит применение специализированного DNS-сервера для трансляции имени ресурса, например, веб-сервера. В зависимости от указанных выше параметров выбора (например, близость узла к клиенту, загрузка узла, наличие в кеше требуемого контента) сервер возвращает различные записи ресурсов A/AAAA, CNAME или NS. Об этих записях мы подробнее говорили в главе 2 «Глобальная система имен».

Так, например, в простейшем случае сервер может вернуть один или несколько IP-адресов оптимального граничного узла. Несколько IP-адресов позволяют осуществлять простейшую балансировку загрузки, когда клиенты будут получать поочередно IP-адреса из множества, тем самым распределяя нагрузку среди нескольких граничных узлов.

В более сложных ситуациях может использоваться многоуровневая трансляция имени, позволяющая распределить процесс принятия решений по маршрутизации запроса между основным DNS-сервером и несколькими специализированными и обладающими более полной информацией DNS-серверами. Для этого используются записи NS или CNAME.

Основной недостаток использования записи NS для «каскадирования» процесса трансляции — то, что число уровней ограничено числом частей самого имени. Например, имя a.b.example.com допускает лишь один уровень «каскадирования» от сервера, отвечающего за example.com, к серверу, авторитетному в зоне b.example.com, который, в свою очередь, вернет IP-адрес a.b.example.com. Поэтому наиболее распространенным является использование CNAME, которое не накладывает такого ограничения.

Например, для получения IP-адреса граничного узла CDN Akamai, обслуживающего сервер www.apple.com, клиенту необходимо обработать три перенаправления: сначала на www.apple.com.edgekey.net., затем на www.apple.com.edgekey.net.globalredir.akadns.net. и, наконец, на e6858.dscc.akamaiedge.net., адрес которого укажет на оптимальный для клиента граничный узел. Этот процесс схематично представлен на рис. 80.

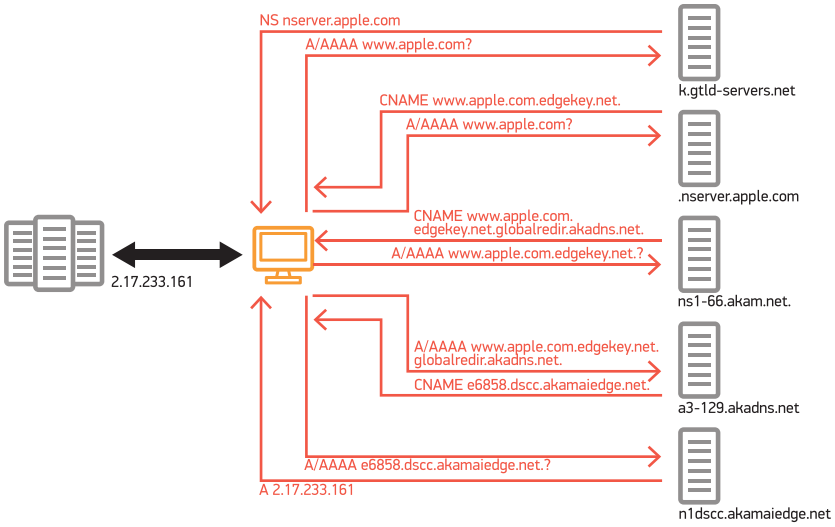

Рис. 80. Каскадная трансляция имени www.apple.com для определения оптимального граничного узла. (Схема указывает принцип работы, так что конкретные имена серверов имен и IP-адреса могут отличаться.)

Как мы видим, DNS-серверы, осуществляющие эти перенаправления, на самом деле являются компонентами системы маршрутизации запросов и генерируют имена в соответствии с алгоритмом поиска оптимального граничного кеширующего узла.

Однако использование DNS для маршрутизации запросов имеет существенные ограничения.

Во-первых, маршрутизация происходит на уровне запрашиваемого доменного имени, хотя часто предпочтительнее была бы маршрутизация на уровне запрашиваемого объекта. Например, объекты контента с различными характеристиками (размер, частота обновления, транспортные характеристики) могут быть размещены на различных кеширующих узлах. И в то же время все они могут быть адресованы различными URL с общим доменным именем. Во-вторых, для определения местонахождения клиента используется IP-адрес DNS-резолвера клиента, а не самого клиента, поскольку DNS-запрос на трансляцию имени поступает именно от резолвера. В некоторых случаях это может привести к существенным ошибкам в определении «ближайшего» граничного узла. Особенно — когда используются резолверы, расположенные вне сети клиента, например, общедоступные резолверы PublicDNS или OpenDNS. Для решения этой проблемы в IETF недавно был документирован механизм[3], позволяющий резолверу при обращении к авторитетному DNS-

[3] RFC 7871 Client Subnet in DNS Queries, https://www.rfc-editor.org/rfc/rfc7871

серверу указать префикс сети клиента (обычно /24 в случае IPv4 и /56 для IPv6). Для передачи этой информации используется специальная опция механизма расширений DNS EDNS0. Однако многие общедоступные резолверы за исключением, пожалуй, PublicDNS (Google) и OpenDNS (Cisco), не поддерживают эту опцию. Отчасти это связано с нежеланием раскрывать информацию ввиду ее частного характера, отчасти с тем, что многие общедоступные резолверы используют глобально распределенную сеть узлов, обеспечивающих эту услугу с помощью технологии anycast. Поэтому запросы приходят от топологически ближайшего к пользователю узла.

Тем не менее данная проблема отчасти остается актуальной, как, впрочем, и проблема недостаточной детализации конкретного запрашиваемого объекта. Для ее решения используются дополнительные методы маршрутизации на транспортном уровне и уровне приложений.

Маршрутизация запросов на транспортном уровне

При использовании этого механизма система маршрутизации анализирует информацию первого пакета запроса клиента на транспортном уровне. Обычно анализом пакета занимается граничный узел, который был выбран с использованием DNS. При этом система получает информацию о фактическом IP-адресе клиента и протоколе приложений, который этот запрос использует. Эта информация позволяет определить оптимальный граничный узел и при необходимости перенаправить запрос.

Маршрутизация запросов на уровне приложений

Анализ запроса на уровне приложений дополнительно позволяет получить информацию о запрашиваемом объекте (или объектах) контента и, соответственно, обеспечить оптимальную маршрутизацию.

Типичным является анализ запрашиваемого URL для более точного определения обслуживающего граничного узла или ближайших кешей более высокого уровня, содержащих запрашиваемые данные. Также часто используется анализ заголовков, таких как Cookie, Accept-language и User-agent, для определения оптимального источника данных. Cookies применяются для идентификации клиента сайта или веб-сессии.

Для перенаправления запроса используется возможность протокола HTTP сигнализировать новый маршрут с помощью кода перенаправления (коды 302 Found или 307 Temporary Redirect). В этом случае ответ содержит новый URL, который приложение должно использовать для доступа к контенту.

Иногда маршрутизация запросов для встроенных объектов, например, изображений, может осуществляться в момент загрузки страницы. В этом случае используется техника «переписывания URL», когда встроенные директивы HTTP переписываются на лету, указывая на оптимальное для данного клиента расположение объектов.

Anycast CDN

Другой подход к определению пограничного сервера, ближайшего к пользователю, состоит в том, чтобы оставить это решение системе интернет-маршрутизации, используя технологию anycast. В соответствии с RFC4786[4] anycast основана на анонсировании IP-адреса сервера (и соответствующего сервиса) в нескольких топологически распределенных местах, так что отправленные дейтаграммы направляются в одно из нескольких доступных мест в соответствии с решениями, принятыми системой маршрутизации.

При использовании anycast всем граничным узлам назначается один и тот же набор IP-адресов, которые анонсируются в системе маршрутизации в местах их конкретного расположения. В этом случае трафик направляется на «ближайший» граничный узел с точки зрения системы маршрутизации, что обычно означает минимальное количество промежуточных сетей (автономных систем в терминах BGP) между пользователем и граничным узлом.

Преимуществом этого подхода является его простота. При достаточном распределении граничных узлов в регионе обслуживания CDN может быть достигнута достаточная близость «ближайшего узла» к пользователю. Одним из основных недостатков anycast является то, что выбор граничного узла в основном находится вне контроля CDN и зависит от топологии межсетевых соединений и политик маршрутизации сетей, участвующих в передаче трафика. Из-за этого управление распределением нагрузки на граничные узлы и другими параметрами, такими как задержка, может быть затруднено.

Google Cloud CDN — это пример CDN, использующей anycast.

P2P CDN

Одной из проблем CDN является плотность покрытия и «последняя миля». Даже при наличии собственной магистрали и тысяч граничных узлов существует практический предел плотности покрытия. Последняя миля, оконечный сетевой сегмент провайдеров широкополосного доступа, в большинстве случаев находится вне контроля CDN.

Для решения этой проблемы иногда используется одноранговый (peer-to-peer, или P2P) подход к доставке контента, т.н. P2P CDN. Он не требует выделенной инфраструктуры и полагается на то, что пользовательские устройства получают доступ к одному и тому же контенту и обмениваются им друг с другом вместо того, чтобы получать его с граничного узла или исходного сервера.

P2P CDN в основном используется для потоковой передачи контента и доставки больших файлов, таких как игровое приложение или обновление ОС. Хотя эту технологию можно использовать для построения автономных CDN, она часто

[4] RFC4786: «Operation of Anycast Services», URL: https://www.rfc-editor.org/rfc/rfc4786

используется в качестве гибридного подхода, когда традиционная CDN доставляет контент на граничный узел, а P2P распределяет его среди потребителей/пользователей сети. Технология, обычно используемая для обмена данными между узлами в P2P CDN, —это WebRTC[5].

Обслуживание статического и динамического контента

Производительность CDN сильно зависит от того, насколько «кешируемым» является исходный контент. Действительно, разница в производительности между попаданием в кеш и промахом может быть значительной. В этом отношении важно понять, как CDN работает со статическим и динамическим контентом.

Статическое содержимое — это содержимое, которое всегда остается одним и тем же, даже если к нему обращаются разные пользователи. Каскадные таблицы стилей CSS, которые применяются для оформления сайта, JavaScript для обеспечения интерактивности, видео и изображения — все это, как правило, не меняется для каждого пользователя для данного URL-адреса и, таким образом, полностью кешируется.

Динамический контент — это контент, который генерируется для пользователей «на лету», когда они просматривают веб-страницу. Динамический контент может быть намного более ресурсоемким, поскольку он может выполнять код и обычно требует запросов к базе данных для создания контента. Многие системы управления контентом (CMS), такие как WordPress, или персональные порталы в области, например, банковского дела или здравоохранения, генерируют динамический контент, который может не полностью кешироваться или вообще не кешироваться.

Традиционно CDN обслуживали статический контент. Запросы динамических элементов, по сути, означали промахи кеша и должны были перенаправляться на исходный сервер. Однако новые методы, такие как интеграция с популярными CMS через API и заголовки управления HTTP, позволяют более эффективно кешировать динамический контент.

Хотя видеопоток выглядит как динамический контент, на самом деле он статичен. Стандартные отраслевые протоколы потоковой передачи, такие как HLS (HTTP Live Streaming)[6] и MPEG-DASH[7], основаны на концепции сегментации. Видеоклип нарезается на отдельные куски или сегменты, каждый из которых длится несколько секунд. Каждый из сегментов имеет отдельный URL и поэтому может кешироваться независимо. Видеопроигрыватель пользователя использует файл «манифеста», связанный с видео, в котором перечислены URL-адреса для всех сегментов видеоклипа. Видеопроигрыватель отправляет HTTP-запросы

5 Real-time communication for the web, URL: https://webrtc.org/
6 https://ru.wikipedia.org/wiki/HLS
7 https://ru.wikipedia.org/wiki/MPEG-DASH

на получение этих сегментов, которые обслуживаются либо из кеша граничного узла (в случае совпадения), либо сначала загружаются с исходного сервера (в случае ошибки).

Инфраструктура с несколькими CDN

Хотя использование CDN улучшает доставку контента по сравнению с одним веб-сайтом, есть несколько соображений, которые поставщик контента должен учитывать при выборе CDN. Ценообразование, дополнительные услуги и общая производительность входят в их число. Но не менее важны географический охват и соображения устойчивости.

Например, одна CDN может обеспечивать превосходное покрытие в отдельных регионах, в то время как другие территории остаются без покрытия. В то же время другая CDN может быть ориентирована именно на эти места, и ее сервис будет дополнять основной. В другом примере провайдер хочет снизить вероятность сбоя, вызванного CDN, и создать план резервного копирования. В обоих случаях использование нескольких CDN вместо одной может стать оптимальным решением.

Типичный способ реализации настройки с несколькими CDN выглядит следующим образом. Предположим, поставщик контента планирует использовать две CDN. Тогда провайдер может использовать DNS в качестве «диспетчера» между ними. Если провайдер поддерживает DNS для своего домена, он может направить разрешение имен на соответствующие серверы имен CDN в циклическом режиме (именно так работает DNS, если присутствует несколько записей для одного и того же имени). Однако такой подход не будет учитывать географическое распределение.

Существуют специализированные провайдеры DNS, которые предлагают услугу «диспетчеризации» на основе геолокации пользователя и даже некоторых общих показателей производительности используемых CDN. Например, компания NS1[8] предлагает интеллектуальные службы DNS, которые направляют пользователя на CDN, оптимальную в данных обстоятельствах (местоположение, производительность и т.д.). Контент-провайдеру нужно просто направить разрешение имен на NS1 или полностью отдать DNS на аутсорсинг для выполнения этой задачи.

Некоторые CDN предлагают использование дополнительной CDN в качестве услуги. Например, клиенты Akamai могут настроить другого провайдера CDN для использования в определенном регионе, при определенной производительности, нагрузке и т.д.

[8] https://ns1.com

Взаимодействие CDN

Эффективность CDN во многом определяется тем, насколько близко точки ее присутствия находятся к потребителю контента. Поэтому многие крупные CDN насчитывают десятки тысяч граничных серверов с глобальным покрытием. Например, на август 2016 года CDN компании Akamai насчитывала более 216 000 серверов, расположенных в 120 странах и работающих внутри более чем 1500 сетей. Но даже такая впечатляющая сеть не может покрыть весь Интернет. В то же время зоны покрытия существующих CDN имеют значительные перекрывающиеся области. Наконец, относительно небольшие CDN, чаще всего развернутые в рамках провайдера интернет-доступа или корпоративной сети, не могут достичь высокой эффективности в силу ограниченного покрытия.

Отметим, что в Интернете глобальная связность обеспечивается путем взаимодействия множества независимых сетей, а не одним суперпровайдером. Представим модель, в рамках которой аналогично происходила бы доставка контента. В рамках этой модели одни CDN могли бы осуществлять «делегирование» доставки другим CDN, тем самым максимизируя общую эффективность.

В этом случае возможным сценарием доставки контента становится следующий процесс:

- Изначальный запрос от потребителя контента (пользователя) принимается авторитетной CDN — то есть CDN, обслуживающей провайдера контента в рамках соответствующего соглашения. Часто такую CDN называют «CDN вверх по потоку» (Upstream CDN, uCDN).
- Авторитетная CDN может обслужить запрос самостоятельно или же перенаправить его другой CDN в случае, если последняя способна выполнить задачу лучше (например, вследствие близости к пользователю). Такую CDN называют «CDN вниз по потоку» (Downstream CDN, dCDN).
- В ответ браузер пользователя запросит контент у dCDN, который, если требуется, будет подкачан из авторитетной uCDN и, если необходимо, от провайдера контента.

Другой пример эффективного взаимодействия CDN — создание федераций. Многие крупные провайдеры широкополосного доступа внедряют собственные CDN. Однако поскольку зона обслуживания таких провайдеров часто включает различные регионы, соответственно, и CDN являются разобщенными. Обмен данными между ними путем создания «федеративной» CDN позволило бы решить эту проблему.

Рабочая группа IETF CDNI (Content Delivery Networks Interconnection) занимается документированием практики и стандартизацией протоколов, необходимых для осуществления обмена данными между автономными CDN.

Общая модель взаимодействия между двумя CDN представлена на рис. 81. Как видно из рисунка, обмен данными осуществляется между соответствующими подсистемами CDN, которые мы обсуждали ранее в этой главе.

Кратко остановимся на интерфейсах этого взаимодействия:

- CDNI Control interface (CI). Управляющий интерфейс, обеспечивающий обмен данными между системами управления CDN. Этот интерфейс необходим для инициализации всех остальных интерфейсов. В этом смысле он также иногда называется Trigger interface.
- CDNI Logging interface (LI). Через этот интерфейс происходит обмен информацией об операциях CDN. Например, взаимодействующие CDN могут использовать его для мониторинга трафика в реальном времени либо асинхронно обмениваться данными для анализа работы и финансовых расчетов.
- Интерфейс маршрутизации запросов отвечает за операции, необходимые для того, чтобы определить, какая CDN (и, возможно, какой граничный сервер) будет обслуживать запрос пользователя. Этот интерфейс состоит из двух взаимосвязанных интерфейсов:

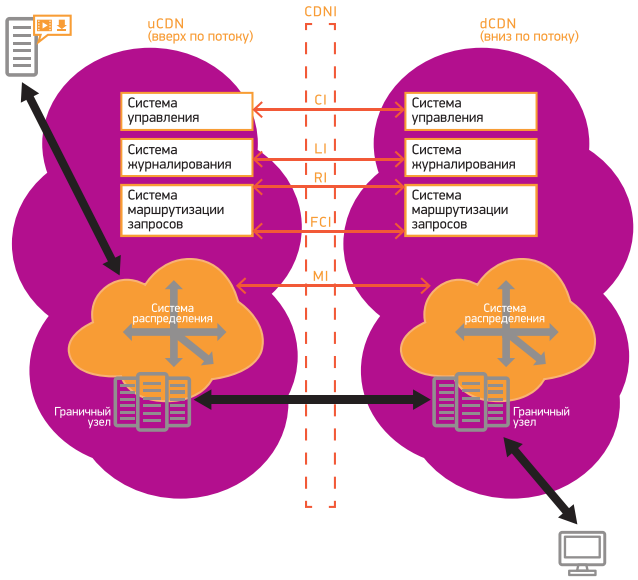

Рис. 81. Структура взаимодействия двух CDN в рамках модели CDNI.

1. CDNI Footprint & Capabilities Advertisement interface (FCI). Через этот интерфейс осуществляется асинхронный обмен информацией, требуемой для маршрутизации запросов, а именно информацией о зоне охвата и возможностях CDN;
2. CDNI Request Routing Redirection interface (RI). Этот интерфейс обеспечивает синхронные операции для определения CDN, отвечающей за доставку контента пользователю в ответ на конкретный запрос.

- CDNI Metadata interface (MI). Этот интерфейс обеспечивает обмен метаданными, определяющими параметры обслуживания контента CDN. В качестве примера таких метаданных можно привести указания по геоблокированию, временные промежутки доступности контента и параметры доступа.

Основные стандарты, определяющие параметры этих интерфейсов и формат данных:

CDN Interconnection Metadata[9]
Эта спецификация определяет формат метаданных и протокол для обмена. Метаданные, связанные с определенным контентом, предоставляют dCDN информацию, необходимую для обслуживания запросов пользователя.

Request Routing Redirection interface for CDN Interconnection[10]
Этот документ определяет формат данных и протокол запросов от uCDN к другой CDN о возможности последней обслужить конкретный пользовательский запрос. Также он определяет ответ обслуживающей dCDN, содержащий информацию о параметрах перенаправления пользовательского запроса.

CDNI Logging Interface[11]
Эта спецификация определяет структуру и протокол обмена информацией об операциях между взаимосвязанными CDN, а также формат файлов регистрационного журнала.

CDNI Control Interface / Triggers[12]
Эта спецификация определяет часть интерфейса CI, позволяющего одной CDN вызвать определенные действия со стороны другой CDN, которой делегирована доставка контента пользователю. Примером таких действий может быть подготовка контента и метаданных или очистка метаданных и кеша.

CDNI Request Routing: Footprint and Capabilities Semantics[13]
Этот документ определяет требования к протоколу FCI и, в частности, тип и формат данных, необходимых для извещения других CDN (выше по потоку) о возможностях и зоне охвата данной CDN.

---

9 RFC 8006: Content Delivery Network Interconnection (CDNI) Metadata, URL: https://www.rfc-editor.org/rfc/rfc8006

10 RFC 7975: Request Routing Redirection Interface for Content Delivery Network (CDN) Interconnection, URL: https://www.rfc-editor.org/rfc/rfc7975

11 RFC 7937: Content Distribution Network Interconnection (CDNI) Logging Interface, URL: https://www.rfc-editor.org/rfc/rfc7937

12 RFC 8007: Content Delivery Network Interconnection (CDNI) Control Interface / Triggers, URL: https://www.rfc-editor.org/rfc/rfc8007

13 RFC 8008: Content Delivery Network Interconnection (CDNI) Request Routing: Footprint and Capabilities Semantics, URL: https://www.rfc-editor.org/rfc/rfc8008

URI Signing for CDN Interconnection[14]
Этот документ описывает, как концепция электронной подписи URI обеспечивает контроль доступа, и предлагает метод подписи URI.

Вопросы безопасности

Вопросы безопасности, относящиеся к различным подсистемам Интернета, таким как DNS или маршрутизация, в полной мере применимы и к CDN. Однако для CDN существуют некоторые особые аспекты, которые мы рассмотрим ниже.

Влияние шифрования данных

Использование протокола TLS/HTTPS для шифрования данных и аутентификации веб-серверов становится все более общей практикой. При этом весь трафик между браузером и сервером зашифрован на уровне приложений. Таким образом, все данные HTTP, включая заголовки, URL и, собственно, сам контент, являются недоступными для промежуточных устройств.

Это, безусловно, представляет проблему для CDN. Отсутствие доступа к этим данным означает, что контент невозможно кешировать, не представляется возможным манипулировать заголовками и проводить анализ трафика для оптимальной маршрутизации запросов. Означает ли это, что сайты, использующие HTTPS, не могут эффективно обслуживаться CDN?

Поскольку соединения HTTPS завершаются на пограничных серверах, сеть CDN должна иметь как публичный сертификат TLS, так и закрытый ключ, соответствующий открытому ключу сертификата. Хотя владелец домена может сам получить сертификат TLS от доверенного удостоверяющего центра (УЦ) и передать его CDN, многие предлагают такую возможность в рамках пакета услуг. На самом деле, многие CDN являются УЦ, способными выдавать такие сертификаты.

Секретный ключ и сертификат TLS, объединяющего параметры ресурса (полное доменное имя и организацию-оператор) с соответствующим публичным ключом, размещаются на каждом граничном узле, обслуживающем определенный контент. Этот процесс схематично показан на рис. 82. Здесь каждый граничный узел может расшифровать передаваемые данные и вновь зашифровать их для передачи от поставщика контента (веб-сайта) и обратно по внутренней сети CDN. При этом могут использоваться другие ключи и схемы шифрования.

[14] RFC 9246: URI Signing for Content Delivery Network Interconnection (CDNI), URL: https://www.rfc-editor.org/rfc/rfc9246

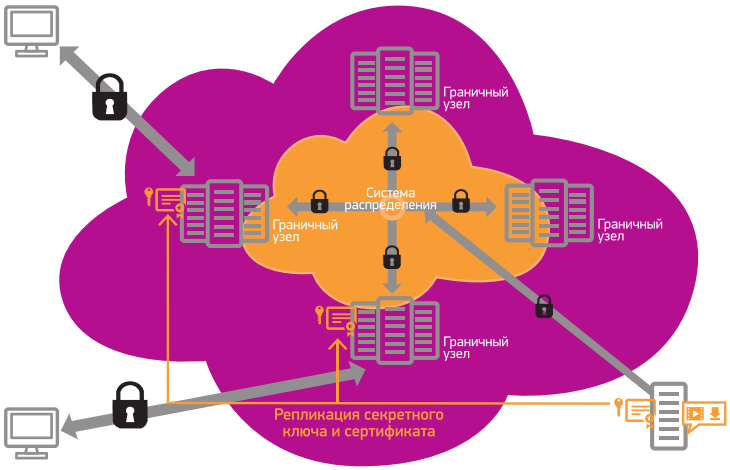

Рис. 82. Размещение TLS-сертификата и соответствующего секретного ключа на граничных узлах, обслуживающих зашифрованный контент.

Однако этот подход несет в себе существенные риски. Секретный ключ хранится в десятках, если не в сотнях тысяч узлов, каждый из которых может иметь уязвимые места, открывающие возможность несанкционированного доступа. Это представляет большую опасность и накладывает серьезные требования на защищенность узлов CDN. Добавим, что каждый граничный узел хранит не один, а множество секретных ключей сайтов, которые данная CDN обслуживает. В рамках совещания IETF96 в июле 2016 года было организовано обсуждение возможных решений данной проблемы на встрече BoF LURK (Limited Use of Remote Keys, Ограниченное использование удаленных ключей). Суть предлагаемой архитектуры заключалась в использовании специального сервера ключей для осуществления криптографических операций. В рамках этой схемы авторизованные граничные узлы обращаются к серверу ключей для шифрования/расшифровывания данных. Даже если один из граничных узлов скомпрометирован, это повлияет только на запросы, обслуживаемые данным узлом, а не на всю систему в целом, как в случае неавторизованного доступа к секретному ключу. Однако эти идеи не получили дальнейшего развития.

Противодействие распределенным атакам

Изначально задачей CDN являлось обеспечение требуемой производительности веб-сервера в периоды пиковой нагрузки. Эффекты Flash crowd и Slashdot — названия одного явления, когда сайт получает слишком много внимания пользователей и не может справиться с нагрузкой, — послужили важным толчком к появлению и развитию CDN. Безусловно, сегодня CDN обеспечивают значительное улучшение и других параметров, таких как задержка, оптимальное использование ресурсов, но распределение пиковой нагрузки по-прежнему остается одной из причин, по которой провайдеры контента используют услуги CDN.

«Пиковая нагрузка» может быть не чем иным как атакой отказа в обслуживании DoS. Эта угроза только увеличивается: мы наблюдаем постоянный рост числа и мощности атак, сила которых сегодня достигает сотен гигабит в секунду. И конечно, CDN играют существенную роль в защите от таких угроз. Благодаря облачной распределенной структуре крупная CDN может «поглотить» и переработать большие объемы трафика, тем самым обеспечивая доступность контента даже во время атак DDoS.

Неслучайно многие компании, предоставляющие услуги CDN, также предоставляют и услуги предотвращения атак DDoS. При этом CDN может предложить нечто большее, чем просто «поглощение» трафика. Современные «митигаторы атак DDoS» обеспечивают так называемый скраббинг (scrubbing) трафика — очистку полезного трафика от вредоносного. Отличить один от другого не так просто, поэтому здесь используются решения, основанные на подходах к защите от спама и вирусов, — анализ структуры трафика на предмет обнаружения известных шаблонов.

Защитный экран веб-приложений (Web Application Firewall, WAF)

Помимо защиты от атак DDoS многие CDN имеют в своем арсенале дополнительные услуги безопасности, такие как WAF. WAF — это защитный экран для приложений HTTP(S). Он применяет набор правил к обмену данными HTTP(S), который охватывает распространенные атаки, такие как межсайтовый скриптинг (XSS) и SQL-инъекция. WAF развернуты на граничных узлах, поэтому они могут останавливать вредоносный трафик ближе к источнику атаки, защищая исходный сервер.

Политика WAF — это свод правил, которые можно настраивать и устанавливать в разных местах CDN. Эти правила включают в себя определение списков разрешенных и заблокированных IP-адресов нежелательных источников запросов, контроль доступа на основе географического положения, правила, основанные на определенных параметрах HTTP-запроса, а также правила ограничения скорости. CDN может предлагать настраиваемые правила, предоставляемые клиентом, вместе с набором правил по умолчанию, например, обеспечивая защиту от 10 основных угроз OWASP[15].

[15] https://owasp.org/Top10

Интернет вещей

> Наиболее совершенными технологиями являются те, которые исчезают. Они все сильнее вплетаются в ткань ежедневной жизни до тех пор, пока не становятся неотличимы от нее.
>
> *Марк Вайзер, «Компьютер XXI века», 1991 г.*

В 1991 году в своем очерке «Компьютер XXI века» Марк Вайзер (Mark Weiser) предвосхитил появление Интернета вещей: «Компьютеры в выключателях, термостатах, музыкальных центрах и духовках уже помогают активировать мир. Эти и другие машины будут соединены между собой во всепроникающую сеть». Он продолжал:

«Присутствие сотни компьютеров в комнате может показаться пугающим на первый взгляд, так же в свое время вселяли страх сотни вольт, бегущих через провода в стенах. Но, как и провода в стенах, эти сотни компьютеров станут для нас незаметны. Люди будут просто использовать их бессознательно для выполнения повседневных задач».

Термин «Интернет вещей» был рожден позже, но Вайзер с удивительной точностью предсказал новый виток эволюции Интернета, когда многочисленные и незаметные устройства, объединенные в единое коммуникационное пространство, помогут перевести наши возможности по сбору, анализу и доступу к данным на качественно новый уровень, преобразуя эти данные в информацию и знание.

«Наиболее важным является то, что вездесущие компьютеры помогут преодолеть проблему информационной перегрузки. Объем информации, доступный нам во время прогулки по лесу, превосходит возможности любой компьютерной системы, и, тем не менее, мы находим прогулку среди деревьев расслабляющей, а работу с компьютером разочаровывающей. Машины, которые встраиваются в человеческую среду, не заставляя нас встраиваться в их собственную, сделают использование компьютера таким же освежающим, как и прогулка в лесу».

В этом разделе речь пойдет о вездесущих компьютерах, которые встраиваются в окружающие нас вещи — от осветительных приборов до одежды и аксессуаров, от кухонных аппаратов до автомобилей — и которые также интегрированы в информационную среду Интернета.

От Интернета сетей к Интернету вещей

История этого витка эволюции Интернета коротка. Говорят, что сам термин «Интернет вещей» впервые появился в 1999 году в презентации соучредителя центра Auto-ID при MIT Кевина Аштона (Kevin Ashton). Центр занимался разработкой технологии RFID и сенсоров, основанных на ней, а в презентации Кевин описал идею использования сенсоров, связанных между собой с помощью Интернета, в процессе производства и распределения продукции.

В то же время концепция использования сенсоров в индустриальных системах не нова. Например, в конце 1970-х некоторые электрические сети использовали коммерчески доступные счетчики, управляемые удаленно по телефонным линиям. В начале 1990-х существовали индустриальные системы, использовавшие достижения в беспроводных технологиях для обмена информацией между отдельными элементами — так называемое взаимодействие M2M (Machine to Machine).

Однако до недавнего времени сенсоры не были встроены в повседневные вещи и были ориентированы на индустриальные решения, а не на потребителя. Для ограниченного взаимодействия между собой они использовали закрытые решения и протоколы. Можно сказать, что они отличались от сегодняшнего феномена Интернета вещей, как арифмометр отличается от компьютера.

Но произошел качественный скачок. Как часто бывает в процессе эволюции, этому способствовали несколько факторов, набравших силу примерно в одно и то же время.

Повсеместная связность. Развитие сетевой инфраструктуры, а также доступность беспроводных решений с использованием мобильной и беспроводной связи делает почти все «подключаемым». Использование общего знаменателя — протокола IP — позволяет мгновенно интегрировать подключенное устройство в глобальную сеть Интернет. Удешевление связи, часто с нулевыми затратами на подключение дополнительного устройства (например, в домашней беспроводной сети), стимулирует проникновение связности в самые невообразимые материальные объекты.

Развитие компьютерных технологий. Закон Мура, предсказывающий экспоненциально возрастающую во времени производительность компьютеров, продолжает править компьютерной индустрией. Мощность процессоров все возрастает, а их размер, как и стоимость, все уменьшаются. Результат — появление миниатюрных устройств-сенсоров с компьютерным интеллектом, доступных потребителю.

Прогресс в области обработки данных. Появление новых алгоритмов анализа данных, а также облачных архитектур для их хранения и обработки при снижении стоимости позволяет существенным образом снизить ограничения на объем собираемой информации. Это, в свою очередь, открывает новые возможности для внедрения большего числа новых сенсоров в новых местах. А создаваемый сенсорами поток данных может быть эффективно обработан и проанализирован, порождая невообразимые объемы новой информации и знаний. Эта информация, в свою очередь, может быть использована для управления самими вещами и для влияния на поведение людей.

Несмотря на то, что сегодня термин «Интернет вещей», или IoT (Internet of Things), материализовался в коммерческих устройствах и системах и используется повсеместно, его общепринятого определения не существует.

Например, некоторые определяют IoT как подключение физических объектов к Интернету и связь их между собой с помощью миниатюрных встроенных сенсоров и проводных и беспроводных технологий для создания экосистемы всепроникающего компьютинга. Другие делают упор на встроенный интеллект в материальных объектах, позволяющий регистрировать изменение их состояния и соответствующим образом реагировать. Отсутствие единой трактовки во многом объясняется тем, что эта область нова и изменчива, к тому же эта тема затрагивает социальные аспекты и имеет как технический, так и философский характер.

Однако можно попробовать выделить несколько общих существенных элементов.

Сенсоры и контроллеры. Звук, движение, наблюдаемые и окружающие объекты, освещенность, температура — эти и другие параметры определяют состояние «вещи» и ее взаимодействие с окружающей средой. Если от «вещи» предполагается действие, она также содержит контроллер — регулятор или управляющее устройство. Так, например, дверь может распознать визитера по биометрическим параметрам и открыть замок, если они соответствуют хозяину жилища.

Отсутствующий пользовательский интерфейс. Большинство «вещей» получают информацию от сенсоров и управляющих серверов. Взаимодействие с пользователем часто происходит опосредованно, через управляющие серверы с интерфейсом порталов или путем использования приложений, с помощью которых пользователь может получить информацию о статусе объектов и задать определенные установки. Умная лампочка осветительной системы Hue компании Philips управляется не привычным диммером или выключателем, а с помощью приложения, установленного на вашем смартфоне или планшете.

Программируемый интеллект. По существу, «вещь», подключенная к Интернету, — это материализованное приложение. И, как в случае обычного приложения, ее функциональность может быть улучшена и расширена. Например, для осветительной системы Hue существует более сотни различных приложений, позволяющих управлять освещением в доме с учетом времени дня, года, музыки, текущей телевизионной программы и т.п. Подключив термостат Nest к Интернету, вы получите доступ к дополнительной функциональности, например, определению оптимального режима в зависимости от прогноза погоды. Проще говоря — границами возможностей является наше воображение.

Связность. Использование Интернета для обеспечения связности «вещей» позволяет им не только обмениваться информацией друг с другом или центральной системой. Интернет обеспечивает доступность «вещей» вне зависимости от вашего расположения. Вы можете управлять отопительной системой вашего дома из салона автомобиля, а климатической системой автомобиля — перед выходом из дома. Открытая коммуникационная инфраструктура Интернета позволяет таким системам обмениваться информацией с другими системами и информационными источниками. Например, система отопления может использовать данные прогноза погоды для выбора оптимального режима.

Автоматизация. Как уже было сказано, индустриальные сенсорные управляющие системы появились задолго до Интернета вещей. Уникальность сегодняшнего явления заключается в том, что оно принесло автоматизацию в массы, позволяя автоматизировать повседневные бытовые задачи. С другой стороны, благодаря открытой коммуникационной инфраструктуре могут быть также решены широкомасштабные задачи — от интеллектуальной транспортной системы до интеллектуального городского освещения.

Сотрудники Глобального института McKinsey выделили несколько областей применения Интернета вещей, благодаря которым можно ожидать дополнительный экономический рост. Наиболее важными, по их мнению, являются производственная и городская инфраструктура, гаджеты, торговля, транспортные системы и, конечно, домашнее хозяйство. Эти области представлены на рис. 83.

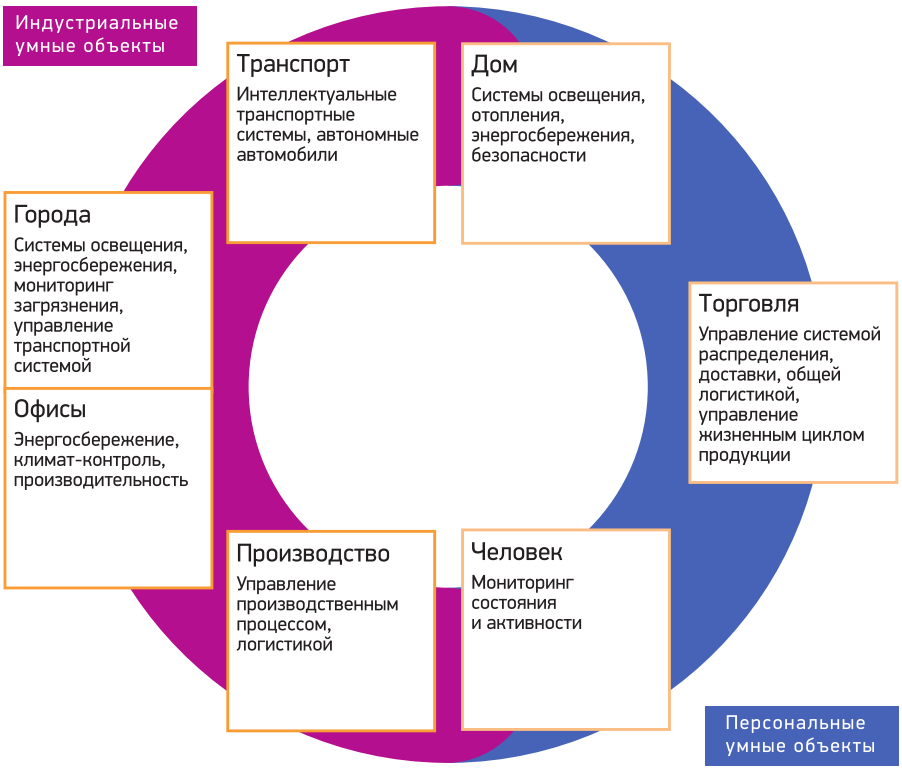

Рис. 83. Области применения IoT согласно данным Глобального института McKinsey.

Архитектурные модели

Мы только что рассмотрели IoT как концепцию и остановились на возможных определениях и отличительных чертах этого явления. Однако в конечном итоге каждый пример IoT — это система, включающая сенсоры и контроллеры, так называемые умные объекты, сети, промежуточные устройства, базы данных, управляющие серверы, порталы и приложения для взаимодействия и управления этими объектами.

Давайте рассмотрим несколько моделей, описывающих взаимодействие между различными компонентами, составляющими Интернет вещей.

Эталонная модель

Одна из моделей была разработана в МСЭ-Т и документирована в рекомендации Y.2060 «Обзор Интернета вещей».

Начнем с того, что рекомендация проводит границу между физическим и информационным мирами, предлагая следующие определения.

Применительно к Интернету вещей устройство означает элемент оборудования, который обладает обязательными возможностями связи и дополнительными возможностям измерения, срабатывания, а также ввода, хранения и обработки данных.

Применительно к IoT «вещи» — это предметы физического мира (физические вещи) или информационного мира (виртуальные вещи), которые могут быть идентифицированы и интегрированы в сети связи. К каждой «вещи» относится информация, которая может быть статической и динамической. Физические вещи существуют в физическом мире, их можно измерить, привести в действие и подключить. Примеры физических вещей — окружающая среда, промышленные роботы, товары и электрическое оборудование. Виртуальные «вещи» существуют в информационном мире, их можно хранить, обрабатывать, а также получать к ним доступ. Примеры виртуальных вещей — мультимедийный контент и прикладное программное обеспечение.

Физическая вещь интегрируется в информационное пространство с помощью устройств, которые взаимодействуют между собой. Устройства могут обмениваться данными с другими устройствами несколькими способами, показанными на рис. 84: с использованием сети связи через шлюз (сценарий a), с использованием сети связи без шлюза (сценарий b) или напрямую, то есть без использования сети связи (сценарий c). Кроме того, возможно сочетание сценариев a и c либо сценариев b и c. Например, устройства могут обмениваться данными с другими устройствами, используя прямую связь через локальную сеть — то есть сеть, обеспечивающую локальное соединение между устройствами и между устройствами и шлюзом, такую как специальная сеть (сценарий c), — и далее обмениваться данными с использованием сети связи через шлюз локальной сети (сценарий a).

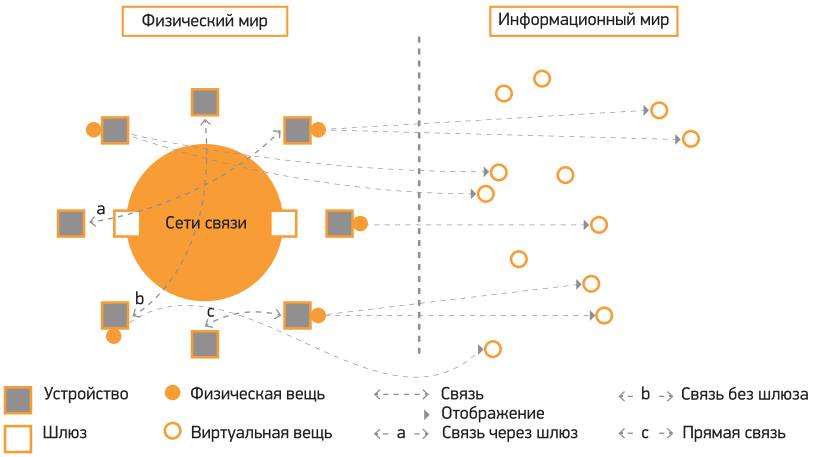

Рис. 84. Технический обзор IoT.
Источник: Рекомендация МСЭ-Т Y.2060 (06/2012)

Хотя на рис. 84 показаны только те взаимодействия, которые происходят в физическом мире (связь между устройствами), взаимодействия происходят и в информационном мире (обмен данными между виртуальными вещами), и между физическим миром и информационным миром (обмен данными между физическими и виртуальными вещами).

МСЭ-Т предлагает четырехуровневую эталонную модель, показанную на рис. 85.

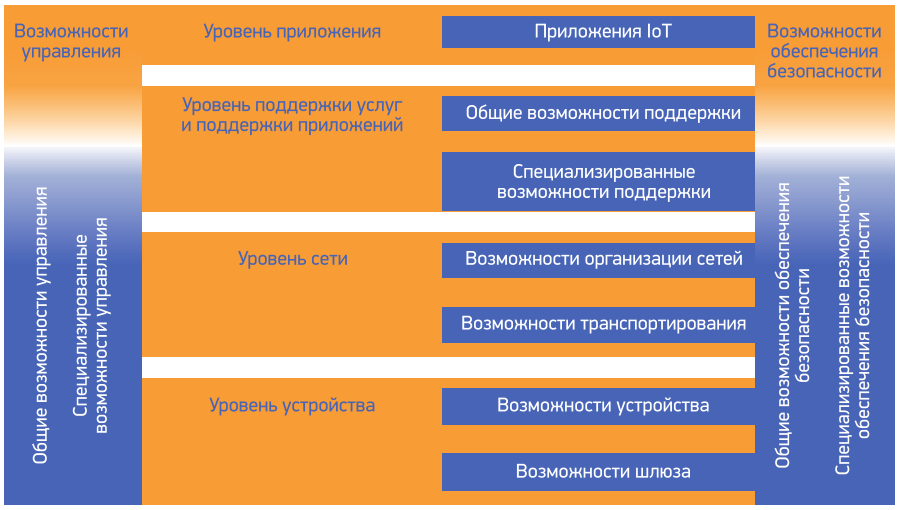

Рис. 85. Эталонная модель IoT МСЭ-Т.
Источник: Рекомендация МСЭ-Т Y.2060 (06/2012)

Самый верхний уровень содержит приложения IoT. Приложения взаимодействуют с уровнем поддержки услуг и поддержки приложений посредством программного интерфейса API.

Уровень поддержки услуг и поддержки приложений включает две группы возможностей:

1. Общие возможности поддержки — это типовые возможности, которые могут использоваться различными приложениями IoT, такие как обработка или хранение данных. Они могут быть активированы специализированными возможностями поддержки, например, для создания других специализированных возможностей поддержки.
2. Специализированные возможности поддержки — это конкретные возможности, которые предназначены для удовлетворения требований разнообразных приложений. В действительности они могут состоять из ряда групп четко определенных возможностей, для того чтобы предоставлять разные функции поддержки разным приложениям IoT.

Уровень сети включает два типа возможностей:

1. Возможности организации сетей: предоставляет соответствующие функции управления сетевыми соединениями, такие как функции управления доступом и ресурсом транспортирования, управление мобильностью или аутентификация, авторизация и учет (AAA).
2. Возможности транспортировки: предназначены для предоставления соединений для транспортировки информации в виде данных, относящихся к услугам и приложениям IoT, а также транспортировки информации контроля и управления, относящейся к IoT.

Наконец, на уровне устройства рассматриваются два вида возможностей в плане взаимодействия с сетью: возможности устройства и возможности шлюза. О возможностях устройства мы поговорим чуть подробнее ниже, в задачу же шлюза входит поддержка различных интерфейсов и трансляция протоколов, например, ZigBee и Bluetooth. При этом предусматриваются две ситуации:

первая ситуация возникает тогда, когда для связи на уровне устройства используются разные протоколы уровня устройства, например, протоколы технологий ZigBee и Bluetooth;

вторая ситуация возникает тогда, когда для связи, требующей и уровня устройства, и уровня сети, используются разные протоколы, например, протокол технологии ZigBee на уровне устройства и протокол технологии 4G на уровне сети.

В IoT минимальным требованием к устройствам является поддержка ими возможностей связи. В Рекомендации устройства подразделяются на несколько

категорий: устройства переноса данных, устройства сбора данных, сенсорные и исполнительные устройства, а также устройства широкого назначения. Они описываются следующим образом.

Устройство переноса данных: подключается к физической вещи и непрямым образом соединяет эту физическую вещь с сетями связи. Примером таких устройств являются активные теги RFID.

Устройство сбора данных: считывающее/записывающее устройство, имеющее возможность взаимодействия с физическими вещами. Взаимодействие может осуществляться непрямым образом с помощью устройств переноса данных или напрямую с помощью носителей данных, подключенных к физическим вещам.

Примером последних являются сканеры штрих- и QR-кодов.

Сенсорное и исполнительное устройство: может получать информацию об окружающей среде, например, о температуре или расположении в пространстве, и преобразовывать ее в цифровые электрические сигналы. Оно может также преобразовывать управляющие сигналы, поступающие от других компонентов IoT посредством сетей, в действия. Как правило, сенсорные и исполнительные устройства образуют закрытые сети, обмениваются друг с другом данными с помощью проводных или беспроводных технологий связи и используют шлюзы для подключения к Интернету.

Устройство общего назначения: обладает встроенными возможностями обработки и связи и может обмениваться данными с Интернетом с использованием проводных или беспроводных технологий. Устройства общего назначения включают оборудование и приборы, относящиеся к различным областям применения IoT, например, станки, бытовые электроприборы и смартфоны.

Примером последних являются сканеры штрих- и QR-кодов.

Сенсорное и исполнительное устройство: может получать информацию об окружающей среде, например, о температуре или расположении в пространстве, и преобразовывать ее в цифровые электрические сигналы. Оно может также преобразовывать управляющие сигналы, поступающие от других компонентов IoT посредством сетей, в действия. Как правило, сенсорные и исполнительные устройства образуют закрытые сети, обмениваются друг с другом данными с помощью проводных или беспроводных технологий связи и используют шлюзы для подключения к Интернету.

На рис. 86 показаны различные типы устройств и их взаимодействие в соответствии с моделью МСЭ-Т.

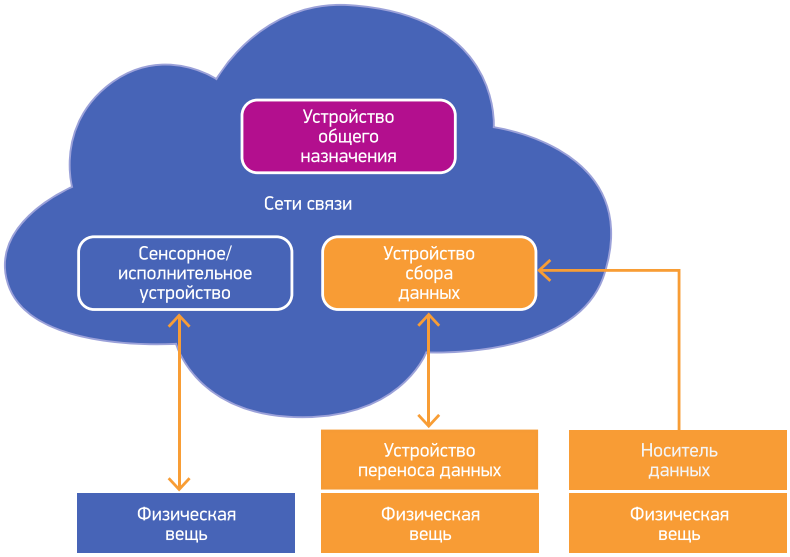

Рис. 86. Типы устройств и их взаимосвязь с физическими вещами.
Источник: Рекомендация МСЭ-Т Y.2060 (06/2012)

Коммуникационные модели IoT

Модели МКЭ-Т предлагают полезную классификацию компонентов экосистемы IoT, однако могут показаться немного абстрактными. Для более практического обсуждения возможных способов взаимодействия между устройствами и другими компонентами системы обратимся к документу IAB «Архитектурные соображения относительно сетей умных объектов», опубликованному в марте 2015 года в виде RFC 7452[16]. Этот документ выделяет несколько коммуникационных моделей, используемых в системах IoT, которые мы рассмотрим ниже.

Взаимодействие устройство–устройство

В рамках этой модели два или более устройств обмениваются между собой данными непосредственно, без участия каких-либо промежуточных элементов. В большинстве случаев связь между устройствами является беспроводной. Для обмена данными используются такие протоколы, как Bluetooth Smart, Z-wave или ZigBee. Простейший пример такой коммуникационной модели — умные лампа и выключатель, которые обмениваются данными о статусе и управляющими командами (рис. 87). Эта модель используется, когда сеть умных объектов является автономной и не требует глобальной связности.

[16] RFC 7452: Architectural Considerations in Smart Object Networking, URL: https://www.rfc-editor.org/rfc/rfc7452

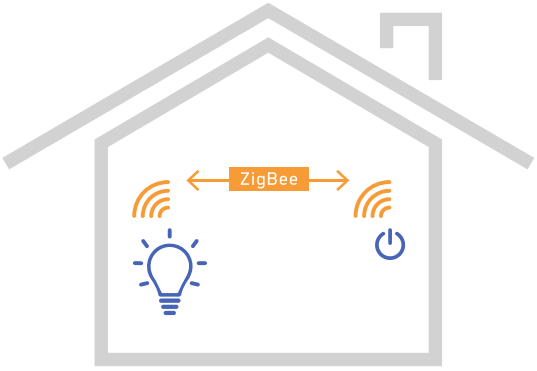

Рис. 87. Пример взаимодействия устройство-устройство.

Поскольку различные коммутационные протоколы несовместимы, устройства такой сети должны использовать один и тот же протокол. Например, семейство устройств с использованием протокола Z-Wave несовместимо с семейством ZigBee-устройств. Это, безусловно, ограничивает пользовательский выбор, хотя в рамках одного семейства, как правило, достигается хорошая совместимость.

Сети могут быть одноранговыми (P2P, peer-to-peer) либо иметь топологию «звезда». На первый взгляд, для решения таких задач, как автоматизация жилища, звезда является наиболее простой топологией с центральной базовой станцией и умными объектами, подключенными к ней. Так обычно реализуется домашняя беспроводная Wi-Fi-сеть. Однако различные помехи, например, трубы и перекрытия, зачастую не позволяют обеспечить связь со всеми объектами из одной центральной точки. Это еще более справедливо в географически распределенных объектах, таких как производство или система водо- или энергоснабжения в масштабах жилого блока или района.

Одноранговые сети, также называемые полносвязными (mesh network) в этом контексте, могут формировать произвольные структуры соединений, и их расширение ограничено только дистанцией между каждой парой узлов. В отсутствии прямой связности два устройства могут обмениваться данными между собой посредством третьего, связанного с первыми двумя. Эта технология позволяет создавать адаптивные беспроводные сети, способные к самоуправлению и самоорганизации.

Взаимодействие устройство-облако

В рамках этой модели устройство непосредственно взаимодействует с провайдером приложения, часто использующего облачные сервисы. Очевидно, что устройство должно поддерживать протокол IP и при этом его связность мало отличается от любого другого устройства — компьютера, смартфона и т.п., подключенного к Интернету. В качестве протоколов нижнего уровня устройства могут использовать IEEE 802.11 (Wi-Fi), что является типичным для домашней

беспроводной сети. Возможно и использование протокола ZigBee IP или Thread, если домашний беспроводной маршрутизатор или базовая станция поддерживают этот стандарт, а сеть поддерживает IPv6.

Пример применения этой модели — уже упоминавшийся обучающийся термостат Nest. Термостат передает данные на удаленный сервер для анализа энергопотребления. Удаленный сервер также является управляющим центром термостата, доступ к которому обеспечивается с помощью приложения, установленного, например, на смартфоне пользователя. Приложение Nest предоставляет пользователю полный доступ ко всем его продуктам Nest. Например, пользователь может изменить температуру или просмотреть потребление энергии у себя дома. Эта модель взаимодействия показана на рис. 88.

Использование провайдера приложений позволяет существенно расширить возможности самой системы в целом. Значительная часть интеллектуальной функциональности может быть реализована провайдером, а у устройства останутся лишь самые необходимые функции — сенсорные, управляющие и коммуникационные. Это также позволяет миниатюризировать устройство, снизить его стоимость и продлить срок службы батареи в случае автономного режима работы.

Например, термостат Nest может функционировать без связи с Интернетом, но его функциональность будет ненамного отличаться от традиционного, пусть и продвинутого термостата.

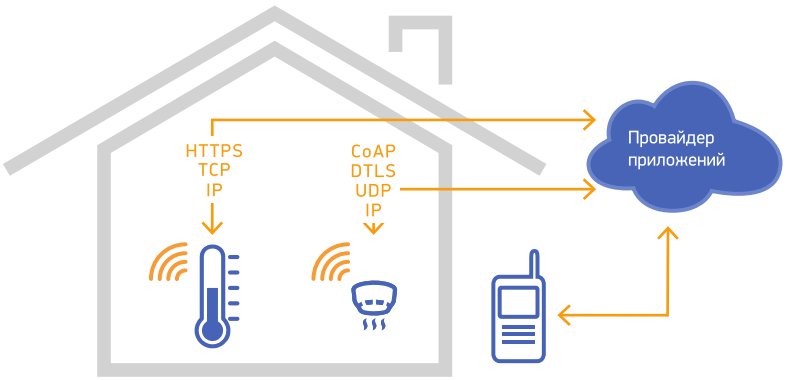

Рис. 88. Пример взаимодействия устройство–облако.

Многие успешные системы IoT представляют так называемые вертикали: когда устройства и облачные услуги провайдера приложений принадлежат одной и той же компании. И хотя порой для связности используются стандартные интернет-протоколы, на более высоких уровнях используются закрытые решения. Иногда провайдер приложений предоставляет API для доступа к облачным

услугам и опосредованного управления устройствами. Это позволяет расширить спектр разработчиков приложений для системы IoT и повысить инновационный потенциал.

Взаимодействие устройство–шлюз

В этом случае устройства взаимодействуют не с провайдером приложений, а с локальным шлюзом, обычно также выполняющим функции шлюза приложений (Application level gateway, ALG). Программное обеспечение шлюза приложений позволяет обмениваться данными с устройствами, управлять ими, обеспечивает безопасность и связь с провайдером приложений. Эта модель представлена на рис. 89.

Наиболее типичный пример реализации данной модели — носимые устройства, такие как фитнес-гаджеты. Сами устройства имеют весьма ограниченную функциональность и зачастую не поддерживают IP. В качестве шлюза обычно используется смартфон с установленным соответствующим приложением. Связь между устройством и смартфоном устанавливается, например, с помощью протокола Bluetooth. В свою очередь приложение смартфона имеет возможность синхронизировать данные с провайдером приложений и получить доступ к дополнительным данным, таким как отчеты и статистика физической активности.

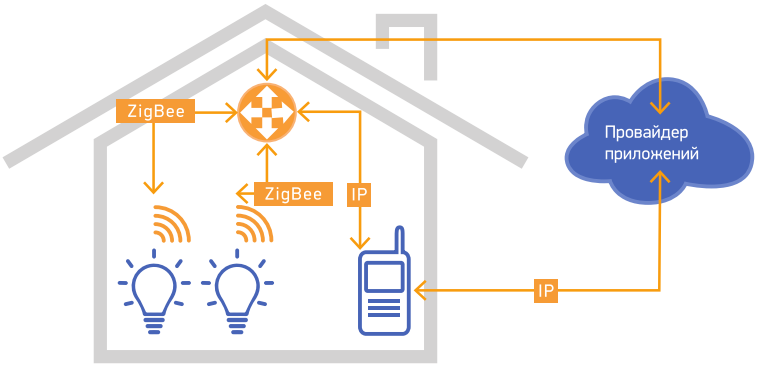

Рис. 89. Пример взаимодействия устройство–шлюз.

Другим типом шлюзов, часто встречающихся в системах автоматизации домашнего хозяйства, являются так называемые хабы, обеспечивающие связь со многими умными объектами в доме. Иногда хабы достаточно интеллектуальны для обеспечения всех функций системы и используют облачные услуги лишь для обеспечения удаленного доступа к системе. Примером такой схемы является система освещения Hue компании Philips. Хаб Hue Bridge обеспечивает управление всеми осветительными устройствами с помощью протокола ZigBee. Hue Bridge также подключается к домашней сети Wi-Fi и доступен через приложение на смартфоне пользователя для задания установок.

Другим примером является хаб SmartThings от Samsung. В этом случае он выполняет дополнительную функцию — обеспечивает обмен данными с устройствами, использующими различные протоколы, например, Z-Wave и ZigBee.

Сети

Одним из важнейших компонентов системы IoT является коммуникационная сеть. Это, как правило, беспроводная сеть, которая должна удовлетворять требованиям и ограничениям устройств, к ней подключенных. В частности:

- Энергосбережение. Многие устройства IoT функционируют в автономном режиме без внешнего источника питания. Срок службы батареи без подзарядки определяется годами, а иногда и десятилетиями.
- Незначительные компьютерные ресурсы, в частности, объем памяти и производительность процессора.

Эти ограничения подключенных устройств и сценарии использования IoT (например, для автоматизации дома и офиса, реализации системы освещения в «умных городах» или системы снабжения в розничной торговле) в свою очередь накладывают на сеть свои ограничения. Ими, в частности, могут являться:

- небольшая пропускная способность;
- возможный высокий процент потери пакетов;
- низкая энергия сигнала;
- асимметричные каналы;
- динамичная топология.

Обычно радиоэлемент (приемно-передающее устройство) является основным потребителем энергии. Также следует иметь в виду, что в энергоэффективных сетях (low power networks) передача данных столь же энергозатратна, как и функционирование в режиме прослушивания. Поскольку энергосбережение является одним из основных требований, накладываемых на устройства IoT, в идеале радиоэлемент должен включаться только в момент передачи данных.

Однако это требование накладывает существенное ограничение на тип сети, которая может быть создана. А именно — в полной мере подойдет только звездообразная сеть с центральным координирующим узлом и периферийными узлами-устройствами. В звездообразной сети радиоэлемент центрального узла включен все время. Это не проблема, поскольку данный узел обычно имеет внешний источник питания. Периферийные узлы, которые питаются от батарей, держат свои радиоэлементы выключенными и включают только в момент необходимости передачи данных. Очевидно, что единственным узлом, которому данные могут быть переданы, является центральный узел.

Звездообразная топология является наиболее простым типом сети, но она имеет существенный недостаток: ограниченный радиус действия. Действительно, для

полноценного функционирования сети все объекты должны находиться в зоне «видимости» центрального узла. Во многих сценариях реализации IoT это не всегда осуществимо.

Для возможности динамического расширения диапазона сети узлы должны выполнять функцию радиотрансляции — принимать и передавать данные между собой, не полагаясь на центральный узел. В этом случае используется так называемая полносвязная сеть (mesh network). За счет ретрансляции протяженность сети может быть значительно увеличена путем добавления дополнительных узлов. Также такая топология предполагает избыточные пути через сеть, обеспечивая повышенную надежность. При выходе какого-либо узла из строя трафик может быть передан в обход.

В такой топологии все узлы могут разговаривать друг с другом, формируя надежную многоскачковую сеть. Но, учитывая требования энергосбережения, радиоэлементы узлов должны управляться таким образом, чтобы они выключались, когда нет трафика, но включались, когда соседи хотят общаться.

Для этого обычно используются два основных подхода — энергосберегающее прослушивание (Low Power Listening, LPL) и синхронизированный по времени ячеистый протокол (Time Synchronized Mesh Protocol, TSMP).

Энергосберегающее прослушивание LPL
При использовании этого подхода узлы работают в импульсном режиме, периодически включая радиоэлемент на непродолжительное время в попытке засечь возможные сигналы о желании передать данные от других соседних узлов. Скважность режима зависит от конкретного типа системы и трафика. Типичным является следующий режим: полсекунды неработы, чередующиеся с несколькими сотнями миллисекунд рабочего состояния.

Устройство, желающее передать данные, сначала посылает стробоскопический сигнал, достаточно продолжительный, чтобы перекрыть период импульсного включения радиоэлемента других узлов. При получении строба при очередном включении радиоэлемента узел оставляет его включенным в ожидании передаваемых данных.

Хотя метод очень прост и не требует явной синхронизации всех узлов сети, он имеет ряд недостатков. Во-первых, стробы пробуждают все узлы, которых они достигают. Во-вторых, чем длительнее период между импульсами включения, тем больше сохраняется энергии, но в то же время тем больше требуется времени на передачу сигнала, поскольку отправителю необходимо сначала передать достаточно долгий стробоскопический сигнал.

Чтобы частично исправить эти недостатки, стробоскопический сигнал содержит идентификатор адресата, таким образом пробуждая только требуемый узел. Также, предполагая постоянство периода импульсов, отправитель может точнее

определить время включения радиоэлемента и, соответственно, непосредственно перед этим послать стробоскопический сигнал. Работа протокола схематически показана на рис. 90(a).

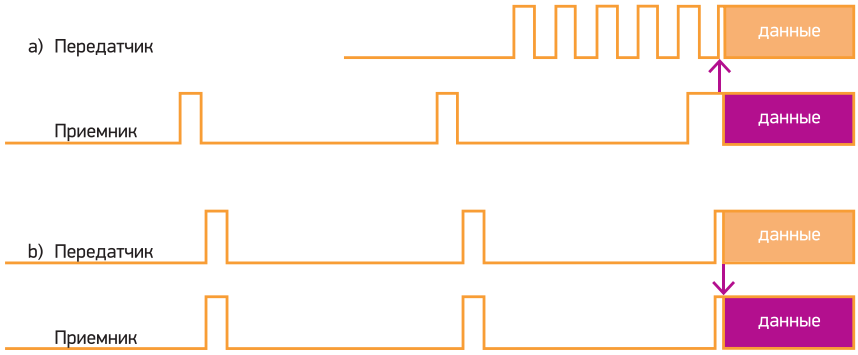

Рис. 90. Работа передающего и принимающего устройства в случае использования LPL (a) и TSMP (b).

Протокол TSMP

Недостатки асинхронных протоколов помогает исправить явная синхронизация всех устройств сети. В этом случае достигается максимальное энергосбережение, так как устройства включают приемники и передатчики на очень короткий промежуток времени. Для синхронизации устройств используется протокол временной синхронизации, например, TSMP (Time Synchronized Mesh Protocol, протокол синхронизации полносвязной сети).

В соответствии с этим протоколом все устройства синхронизированы с точностью до 50 мкс. В начале каждого временного слота продолжительностью 10 мс все устройства кратковременно включают радиоэлементы. Если устройству необходимо передать данные в текущем временном слоте, оно должно начать передачу в течение 100 мкс с его начала. Таким образом, в общем случае принимающие устройства должны держать свои радиоэлементы включенными не дольше 100 мкс на каждые 10 мс. Работа протокола схематически показана на рис. 90(b).

Протокол TSMP требует централизованного управления сетью. Он был разработан для индустриальных областей внедрения IoT и не используется в сетях автоматизации дома и офиса.

Стандарты

В зависимости от характера применения систем IoT для связи объектов используются различные сетевые технологии и протоколы. Отчасти это связано с развитием проприетарных решений, созданием так называемых вертикалей,

когда ноу-хау является частью конкурентоспособности системы. Кроме того, системы IoT значительно различаются по своим характеристикам и требованиям к автономности функционирования, географической протяженности, объемам передаваемых данных и т.п. Например, радиус действия систем автоматизации дома и офиса, как правило, не превышает нескольких десятков метров, в то время как системы, внедряемые в «умных городах» (например, системы освещения, контроля транспорта и энергосбережения), требуют дальности передачи, измеряемой десятками километров.

Мы уже кратко остановились на некоторых стандартах, используемых в IoT. Основные усилия здесь направлены на разработку беспроводных технологий физического и канального уровней, которые бы удовлетворяли требованиям и ограничениям, связанным с IoT. Все беспроводные протоколы, за исключением мобильной связи, используют нелицензируемый спектр. На рис. 91 представлены стеки нескольких наиболее популярных протоколов и технологий. Поговорим о них подробнее.

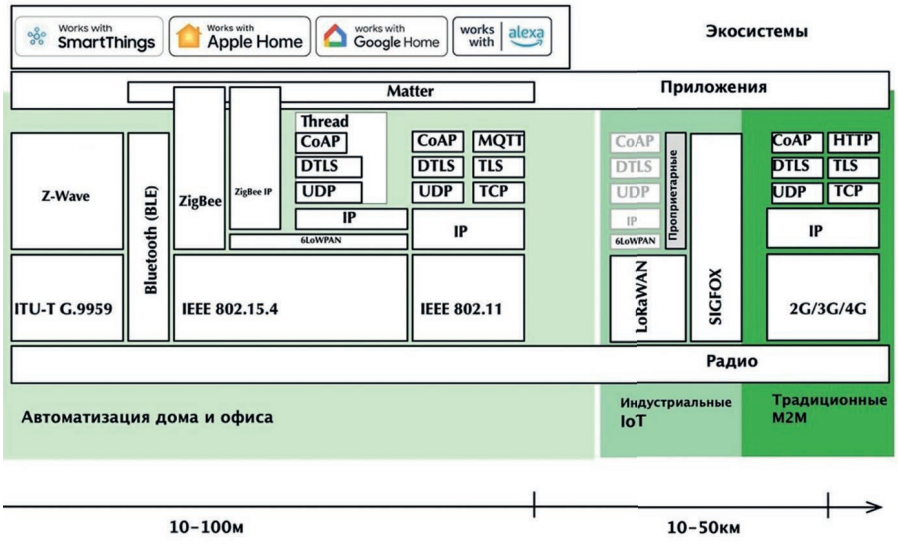

Рис. 91. Стеки популярных протоколов IoT.

Bluetooth

Bluetooth, пожалуй, является самым старым и знакомым протоколом для связи между различными устройствами на коротких расстояниях. Изначально разработанный компанией Ericsson в 1994 году в качестве беспроводной альтернативы серийному кабелю RS-232, этот протокол используется в широком спектре компьютерных устройств. Протокол был стандартизован IEEE и в настоящее время обслуживается организацией Bluetooth Special Interest Group.

Этот протокол доминирует в системах управления и обмена информацией с различными гаджетами — от фитнес-сенсоров до традиционных наушников и спикеров. Обмен данными обычно происходит между гаджетом и смартфоном, выполняющим роль шлюза. Новая версия протокола — Bluetooth Low-Energy (BLE), или Bluetooth Smart, — является важным протоколом для приложений IoT. Обеспечивая диапазон действия, аналогичный традиционному Bluetooth, BLE позволяет значительно сократить потребление электроэнергии.

При поддержке профиля Internet Protocol Support и адаптационного уровня 6LoWPAN, о котором мы поговорим чуть позже, устройства способны непосредственно использовать протокол IPv6. Благодаря этому системы IoT могут быть напрямую подключены к Интернету для управления умными объектами.

IEEE 802.11

Беспроводные сети на основе протоколов семейства IEEE 802.11 составляют основу Wi-Fi-хот-спотов, офисных и домашних сетей. Этот стандарт поддерживается практически всеми переносными компьютерными устройствами — от смартфона до ноутбука и персонального компьютера. Скорости в 54 Мбит/с являются вполне обычными (IEEE 802.11a/g), но все больше сетей поддерживают более высокую пропускную способность в сотни Мбит/с (IEEE 802.11n/ac).

Энергосбережение не является отличительной чертой этих протоколов, поэтому они могут быть использованы в системах, где устройства подключены к источнику питания. Однако версия протокола IEEE 802.11ah является энергосберегающей и была специально разработана для систем IoT.

IEEE 802.15.4

Стандарт IEEE 802.15.4 разработан и поддерживается рабочей группой IEEE 802.15. Он определяет физический уровень и уровень управления доступом к среде (Medium Access Control, MAC) для энергоэффективных беспроводных персональных сетей небольшой дальности действия. Максимальная скорость передачи, определенная стандартом, — 250 кбит/с, а выходная мощность равна 1 мВт. Номинальный радиус действия — несколько десятков метров.

Этот стандарт является «строительным блоком» для нескольких других протоколов IoT, таких как ZigBee, ZigBeeIP и Thread, определяющих верхние слои стека вплоть до уровня приложений.

IEEE 802.15.4 может использоваться совместно со стандартом 6LoWPAN, который определяет адаптационный уровень IP. В этом случае в качестве верхних уровней стека могут применяться стандарты IETF — UDP, DTLS, CoAP, а устройства могут быть напрямую подключены к Интернету с глобальной связностью.

6LoWPAN

Стандарт 6LoWPAN разработан и поддерживается IETF. Он определяется спецификациями RFC 4944[17], RFC 6282[18] и RFC 6775[19]. С помощью этого стандарта пакеты IPv6 могут передаваться по сетям IEEE 802.15.4. Дело в том, что стандартные пакеты IPv6 слишком велики для энергоэффективных беспроводных сетей IoT, таких как IEEE 802.15.4.

Стандарт IEEE 802.15.4 предоставляет всего 81 байт для протоколов верхнего уровня, что значительно меньше максимального размера пакета (Maximum Transmission Unit, MTU) IPv6, не требующего фрагментации, — 1280 байт. Поскольку в стандарте IPv6 промежуточные устройства не осуществляют фрагментацию, эту функцию должен выполнять шлюз между сетью IEEE 802.15.4 и Интернетом. Стандарт определяет адаптационный уровень между IEEE 802.15.4. и IP, обеспечивающий необходимую фрагментацию и сбор пакетов. Но это еще не все. Только заголовок IPv6 занимает почти половину доступного места (40 байт), что, с учетом заголовков фрагментации и накладных расходов протоколов более высокого уровня, практически не оставляет места для полезных данных. Решением является компрессия заголовка, определенная в RFC 4944 и RFC 6282. Предложенные алгоритмы позволяют сжать заголовки IPv6 и UDP до 12 байт.

ZigBee

ZigBee — один из популярных протоколов, применяющихся в системах домашней и офисной автоматизации, например, в системах освещения. Система Hue компании Philips использует этот протокол для обмена данными с осветительными приборами и шлюзом Hue Bridge.

ZigBee использует протокол IEEE 802.15.4 для нижних уровней стека, но определяет собственное шифрование и маршрутизацию данных. Нужно заметить, что протокол ZigBee также обслуживает уровень приложений и, соответственно, имеет несколько профилей для различных сценариев применения: домашняя автоматизация, энергетика, здравоохранение и т.д. Новый стандарт, ZigBee 3.0, унифицирует обмен данными между этими кластерами приложений, облегчая интеграцию различных систем.

Для некоммерческого использования спецификации ZigBee доступны бесплатно. Для производителей требуется вступление в ассоциацию ZigBee Alliance.

---

[17] RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks, URL: https://www.rfc-editor.org/rfc/rfc4944

[18] RFC 6282: Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks, URL: https://www.rfc-editor.org/rfc/rfc6282

[19] RFC 6775: Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs), URL: https://www.rfc-editor.org/rfc/rfc6775

Как и у многих протоколов локальных энергоэффективных сетей, радиус действия ZigBee не превышает нескольких десятков метров, но может быть расширен за счет промежуточных устройств. Скорость передачи данных также невысока — 250 кбит/с.

ZigBee IP

ZigBee IP является версией протокола ZigBee, поддерживающей протокол IPv6 на сетевом уровне. Для этого ZigBee использует адаптационный уровень 6LoWPAN. Применение ZigBee IP позволяет осуществить сквозную адресацию устройств IoT-системы, обеспечив их непосредственную связность с Интернетом.

Z-Wave

Так же, как и ZigBee, Z-Wave является одним из наиболее популярных технологий систем IoT для автоматизации дома и офиса. Основу технологии составляет радиоэлемент и протокол компании Sigma Designs.

В качестве протокола нижних уровней Z-Wave использует ITU-T G.9959, реализуя верхние уровни вплоть до приложений стеком протоколов от Sigma Designs. Стандарт Z-Wave поддерживается ассоциацией Z-Wave Alliance. Z-Wave использует частоту 900 МГц и работает в топологии полносвязной сети.

Для разработчиков и производителей чипов Sigma Designs лицензирует дизайн, программное обеспечение стека и программный интерфейс. Спецификации Z-Wave доступны без роялти, на основе модели RAND (Reasonable and Non-discriminatory, https://ru.wikipedia.org/wiki/Reasonable_and_Non-Discriminatory).

Thread

Thread был впервые анонсирован в 2014 году коалицией компаний, включающих Nest Labs от Google, Samsung Electronics и ARM. Разумеется, первым устройством, поддерживающим этот стандарт, был термостат Nest, но число совместимых систем неуклонно растет.

Thread, пожалуй, самый открытый стандарт из всех перечисленных. Как и ZigBee, он использует стандарт IEEE 802.15.4 для нижних уровней стека, а также адаптационный уровень 6LoWPAN для поддержки IPv6. Другими словами, все устройства Thread используют IPv6 в качестве сетевого протокола — а значит, эти устройства органично интегрированы в домашнюю Wi-Fi-сеть и Интернет.

Thread позволяет создать одноранговую сеть взаимодействующих друг с другом устройств. Это означает, что все устройства Thread обмениваются данными друг с другом (вместо связи только с базовой точкой доступа, как в Wi-Fi-сетях), что обеспечивает повышенную надежность и более широкий охват. Хотя бы одно из устройств должно выполнять функции «граничного маршрутизатора», обеспечивая связь этой одноранговой сети с домашней сетью.

Matter

Одной из проблем современных IoT является совместимость на уровне приложений. Обратите внимание на знаки «работает с...» на многих устройствах умного дома сегодня. «Работает с Apple Home», «работает с SmartThings», «работает с Alexa», «работает с Google Home» - каждый такой значок указывает, какую экосистему умного дома поддерживает это устройство. В рамках одной экосистемы устройства работают безупречно, но попробуйте подключить «чужестранца», и вы сразу столкнетесь с проблемой совместимости. Скорее всего, интегрировать такое устройство не получится, и вам придется управлять им специально для него созданным приложением.

С пользовательской точки зрения такая ситуация нежелательна, да и для производителей устройств и даже создателей экосистем она не оптимальна. Поэтому несколько лет назад Apple, Google, Amazon, Samsung и другие объединились через Альянс стандартов связности (Connectivity Standards Alliance, CSA; который ранее назывался Альянсом ZigBee), чтобы попытаться создать единый стандарт, который позволил бы одному устройству работать через множественные экосистемы. Эта работа завершилась в 2022 году созданием стандарта под названием Matter.

Matter работает поверх существующих сетевых и транспортных протоколов, поддерживая Thread, Wi-Fi, Ethernet и BLE. С другой стороны, Matter определяет стандартный набор команд для управления и интеграции устройств в экосистемы IoT.

Таблица 8. Сравнительные характеристики популярных технологий беспроводной связи в системах IoT

	Bluetooth BLE	ZigBee	Z-Wave	Thread	IEEE 802.11
Дальность	50–150 м	10 м	30 м	30 м	50 м / 1 км(802.11ah)
Максимальное количество устройств сети	8 (пиконет)	65 000	232	300	2007
Скорость передачи	1 Мб/с	40–250 кб/с	9.6–100Кб/с	250 Кб/с	50–200 Мбит/с 100 кбит/с (802.11ah)
Частота	2,4 ГГц	915 МГц / 2,4 ГГц	900 МГц	2,4 ГГц	900 МГц (802.11ah) / 2,4 ГГц /5 ГГц
Тип сети	звездообразная	полносвязная, древовидная, звездообразная	полносвязная	полносвязная	звездообразная

Sigfox

Французская компания Sigfox была основана в 2009 году для построения инфраструктуры для различных распределенных приложений IoT, таких как системы освещения, индустриального контроля и т.п. Разработанная технология и инфраструктура также получила название Sigfox. Однако, истратив 350 миллионов евро инвестиций, Sigfox зарегистрировала всего около 20 миллионов подключений, и в январе 2022 года вынуждена была подать заявление о неплатежеспособности, не перенеся экономических последствий COVID-19. Спустя

несколько месяцев она была приобретена сингапурской компанией UnaBiz, которая с тех пор является владельцем технологии Sigfox 0G. Эта технология используется более чем 70 0G-операторами по всему миру.

Технология Sigfox 0G основана на сетевом протоколе с низким энергопотреблением (LPWA) и определяет полный пакет — от радиопротокола, программного обеспечения коммуникационных элементов устройств до коммуникационной инфраструктуры и облачного хранилища данных и доступа к данным систем IoT. Единственным более или менее открытым элементом системы является лицензирование радиотехнологии различным производителям. Например, STMicroelectronics, Atmel и Texas Instruments производят радиоэлементы для сети Sigfox 0G.

Sigfox 0G использует узкий диапазон радиоспектра, что обеспечивает лучшую защиту от шума даже при маломощной передаче. Оконечные устройства недорогие, особенно по сравнению с весьма дорогостоящими базовыми станциями, обслуживающими сеть.

Наибольшее покрытие Sigfox 0G обеспечивает в странах Западной Европы, см. рис. 92, хотя сегменты сети разворачиваются глобально, включая США, Новую Зе-

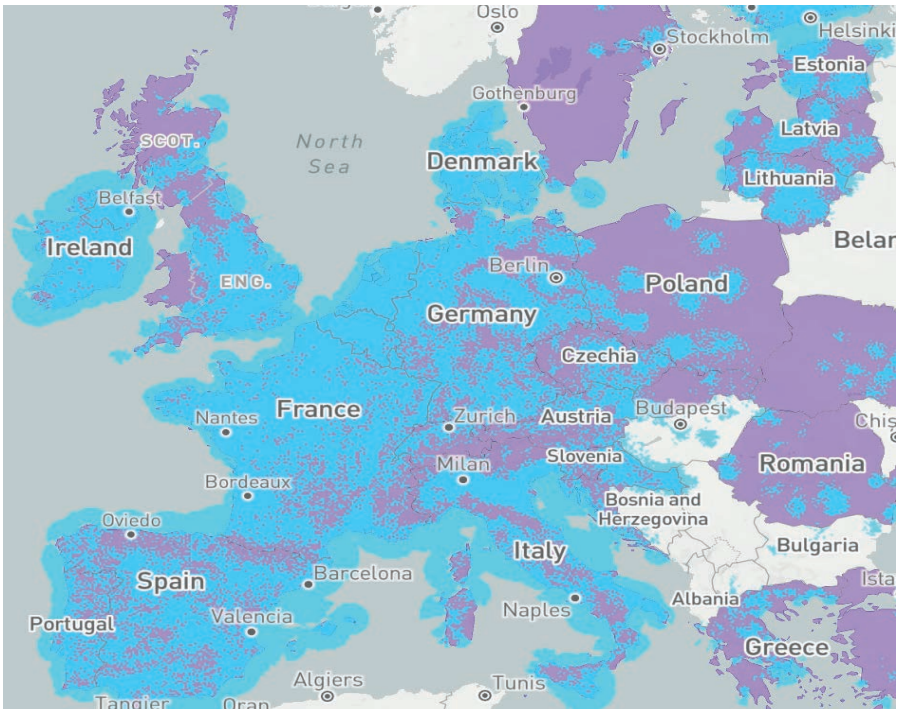

Рис. 92. Зона покрытия сетей Sigfox. Голубым цветом показано текущее покрытие, фиолетовым — области, где в настоящее время происходит развертывание технологии.
Источник: https://www.sigfox.com/coverage

ландию, Австралию, Бразилию и Оман. UniBiz работает с региональными сетевыми операторами, которые, собственно, и разворачивают эту технологию, выплачивая UniBiz роялти за ее использование. Другими словами, они выступают как реселлеры сетевых услуг и услуг приложений Sigfox 0G.

LoRaWAN

LoRaWAN отчасти напоминает Sigfox, хотя есть и существенные отличия, как в плане технологии — радио LoRa использует широкий спектр (обычно более 125 кГц), — так и в отношении бизнес-модели. LoRa является технологией радиомодуляции для сетей большой дальности с низким энергопотреблением и низкой скоростью передачи данных. Основные особенности данной технологии — большое количество устройств, которые могут работать в рамках одной локальной сети, и относительно большая дальность, которая может быть покрыта одним маршрутизатором LoRaWAN. Один шлюз способен координировать около 20 000 узлов в диапазоне 10–30 км.

LoRaWAN является более открытой системой. Спецификации LoRaWAN свободно доступны — любой производитель сетевого оборудования может начать выпускать модули или шлюзы для этой системы. Более того, сетевой оператор может создать собственную инфраструктуру, включая систему сбора, анализа и доступа к данным. Единственным ограничением является то, что радиоэлемент LoRa основан на чипе компании Simtech.

Разработкой и сопровождением стандартов, а также сертификацией оборудования занимается отраслевая ассоциация LoRa Alliance с достаточно широким спектром организаций-членов[20].

Стандарты сотовой связи

Технологии передачи данных сотовой связи 2G/3G/4G/5G могут использоваться для приложений IoT, требующих обмена информацией на больших расстояниях с использованием существующей инфраструктуры. Эти сети способны передавать большие объемы данных, особенно 5G, однако потребляемая мощность является недопустимо высокой для многих применений. Впрочем, для систем с устройствами, осуществляющими нечастый кратковременный обмен данными, эти технологии вполне приемлемы.

Вопросы безопасности и защиты персональных данных

Вместе с ростом числа и типов устройств, подключенных к Интернету, растут и риски, связанные с безопасностью и защитой частной жизни. Это особенно справедливо для IoT: окружающие нас вещи могут быть использованы не по назначению и в преступных целях, их функциональность может быть изменена вплоть до отказа работы. В то время как умные объекты все теснее вплетаются в нашу жизнь, делая нас все более зависимыми от них, вопросы обеспечения защищенности систем IoT, персональных данных становятся как никогда актуальными и составляют важный элемент нашей собственной безопасности и защищенности частной жизни.

[20] https://lora-alliance.org/member-directory

Несколько факторов усугубляют ситуацию:

- Индустрия умных объектов достаточно молодая. Многие компании являются либо стартапами, либо берут свои истоки в отраслях, отдаленных от компьютеров и Интернета. Это зачастую приводит к недостаточному знанию и опыту в вопросах компьютерной безопасности. Например, использование открытых каналов для обмена данными между устройствами является одним из встречающихся уязвимых мест таких систем.
- Основной упор делается на функциональность устройств и системы в целом. Учитывая желание минимизировать стоимость устройств, это зачастую приводит к недостаточному вниманию к вопросам безопасности.
- Масштаб внедряемых систем IoT существенен. Домашняя, офисная или индустриальная сеть теперь должна обслуживать на порядки больше устройств. Более того, однотипность этих устройств значительно усиливает эффект обнаружения уязвимости в одном из них.
- Некоторые устройства не имеют возможности автоматического обновления программного обеспечения с целью закрытия уязвимых мест. Учитывая продолжительный срок жизни многих устройств, отсутствие такой функциональности представляет существенный риск. Возьмем, например, инцидент с маркой Джип[21], когда была открыта уязвимость, позволявшая атакующему удаленно получить контроль над важными функциями автомобиля, включая тормоза. Компании Fiat Chrysler пришлось отозвать 1,4 миллиона машин для обновления программного обеспечения. Учитывая неудобства, которые эта операция доставляет многим владельцам, можно предположить, что в значительной части автомобилей уязвимость осталась незакрытой.
- Все больше окружающих нас вещей используют Интернет для расширения своей функциональности. Становится все сложнее приобрести «вещь», которая бы не подключалась к Интернету.
  Все более важные жизненные аспекты зависят от правильного функционирования таких систем. Например, «глюк» с термостатом Nest[22] чуть не заморозил владельцев этого устройства.
- Многие сенсоры производят сбор весьма конфиденциальных данных, предоставляющих информацию о наших привычках, поведении, нахождении, могут прослушивать наши разговоры и делать видеозаписи. Например, устройства SmartTV компании Samsung могут управляться голосовыми командами. Проблема заключается в том, что для этого телевизор пересылает услышанную речь в Samsung для анализа на предмет возможных команд[23]. Разумеется, это зачастую не только команды, но и просто подслушанный разговор. Насколько хорошо защищены эти данные, и кто имеет к ним доступ?

---

21 https://www.wired.com/2015/07/jeep-hack-chrysler-recalls-1-4m-vehicles-bug-fix

22 https://www.nytimes.com/2016/01/14/fashion/nest-thermostat-glitch-battery-dies-software-freeze.html?_r=0

23 https://www.cnet.com/news/privacy/samsungs-warning-our-smart-tvs-record-your-living-room-chatter

Причиной многих из этих проблем являются не IoT как таковой, а то, что цифровой мир и Интернет все плотнее интегрируются с физическим миром. Все больше персональных и конфиденциальных данных хранятся в «облаках», а мы все более зависимы от умных полезных устройств, приложений, Интернета. IoT, безусловно, делает некоторые из этих проблем значительнее.

Защищенность системы — не бинарное состояние: степени безопасности представляют собой широкий спектр. Защищенность системы также зависит и от характера угроз. Все эти факторы меняются во времени. Будем надеяться, что по мере того как отрасль становится более зрелой, безопасность IoT будет обеспечиваться на адекватном уровне.

Заключение

Взгляд в будущее открывает неполную и порой не совсем верную картину. Несмотря на то, что технология является мотором изменений в Интернете, успех того или иного нововведения определяется социально-экономическими факторами.

Как и наш физический мир, развитие Интернета подчиняется законам естественного отбора. Разрабатываются новые технологии и основанные на них решения — некоторые тихо умирают, некоторые распространяются с эпидемической скоростью. Благодаря глобальному покрытию и миллиардам участников в Интернете постоянно происходят мутации — новые приложения, протоколы, улучшенные версии существующих, — которые мгновенно становятся доступными в глобальном масштабе. Что-то приживается и пускает корни, что-то — нет.

Спроектировать Интернет невозможно. Его развитие определяется взаимодействием различных тенденций и интересов, а не согласованным планом или архитектурной моделью или даже технологией.

И в то же время этот процесс не является полностью случайным. Мы можем увидеть очертания будущего в технологиях и приложениях настоящего. Не думаю, что мы сильно ошибемся, если предположим, что тенденция увеличения пропускной способности и качества доступа к информации и, что более важно, к знанию будет продолжаться, превращая Интернет в гигантский суперкомпьютер. «Сеть — это компьютер» — слоган уже несуществующей компьютерной компании Sun Microsystems сегодня верен как никогда. Спроектировать Интернет невозможно. Но каждый из нас вносит вклад в его будущее.