

Управляется программно: Облачная сеть

Продолжим наш разговор про "программно-конфигурируемые сети", они же SDN. Тема сложная, но важная в развитии телекоммуникаций, поскольку потенциально меняет подход к построению сетей вообще. И даже может изменить бизнес-ландшафт отрасли. Ключевое слово здесь - потенциально.

Материал сей является отчетным по гранту полученным ООО "ФРИИ Вай-Фай" от "[Фонда содействия инновациям](#)". И он об исследовании по теме "Программный комплекс мониторинга и управления распределенными сетями передачи данных для организации систем условного доступа в Интернет".

Для начала определимся с термином "управление", потом порассуждаем об объекте управления и целях "управления". Это в качестве введения.

Введение в управление

Собственно, в "Теории управления" (имеется в виду "наука о принципах и методах управления различными системами, процессами и объектами" - она же "[Control Theory](#)" - далее будем писать просто ТУ), главной целью является понимание принципа функционирования систем (любых - инженерных и/или социальных) для последующего решения прикладных задач моделирования различных систем, процессов и объектов, позволяющего существенно увеличить возможности автоматизации человеческого труда.

Проще говоря, прикладная цель управления (в данном случае) - все автоматизировать и таким образом, чтобы системы могли функционировать эффективнее, продуктивнее и желательно при минимальном вмешательстве человека.

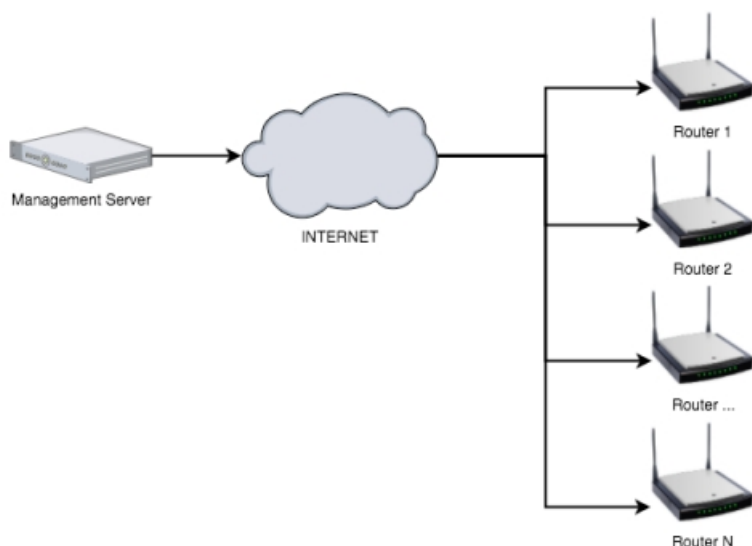
Существует огромное количество математических методов описания задач и, соответственно, и не меньшее количество способов решения этих задач. Но обычно все сводится к последовательности, для уже определенного на этапе постановки задачи объекта управления, последовательности:

1. Сбор и обработка информации об объекте управления.
2. Анализ, систематизация, синтез.
3. Постановка на этой основе целей. Выбор метода управления, прогноз.
4. Внедрение выбранного метода управления.
5. Оценка эффективности выбранного метода управления.

Но все же главное - это объект управления, потому что имеется разница между управлением, например, клапаном холостого хода инжекторного двигателя, и таким сложным и комплексным объектом, как сеть передачи данных.

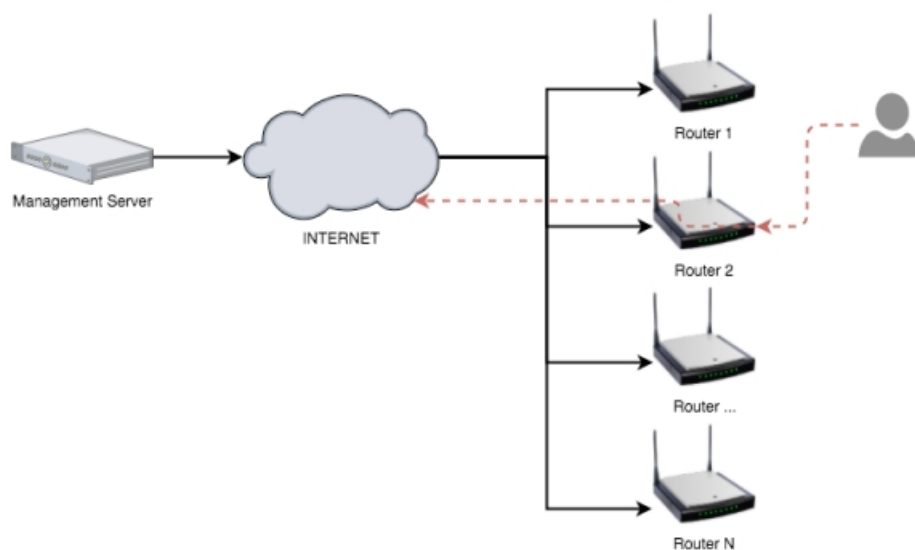
Особенно, если сеть распределена географически, использует наложенные каналы передачи данных и оказывает услуги "банального доступа" в интернет, но по определенным условиям.

Собственно, управлять именно такой сетью и планировалось, исходя из условий технического задания. И в общем виде эта сеть выглядит так:



То есть, у нас имеется N элементов сети (устройств), которые подключаются к "облаку интернет", с произвольными сетевыми характеристиками, где мы понятия не имеем о параметрах каналов подключения каждого устройства и никак не можем на них влиять. Каждый сетевой прибор находится в зоне ответственности клиента/заказчика услуги, а каналы подключения принадлежат "третьим лицам" в лице операторов связи, которые продали услугу "доступ в интернет" собственно заказчикам.

Усложним схему, изобразив на ней "гостя Wi-Fi сети", который подключается к управляемому устройству, но пользуется услугами доступа по сути стороннего оператора связи, с которым нет юридических отношений и на который мы никак повлиять не можем. И даже больше, таких операторов может быть больше одного, если используются резервный оператор.



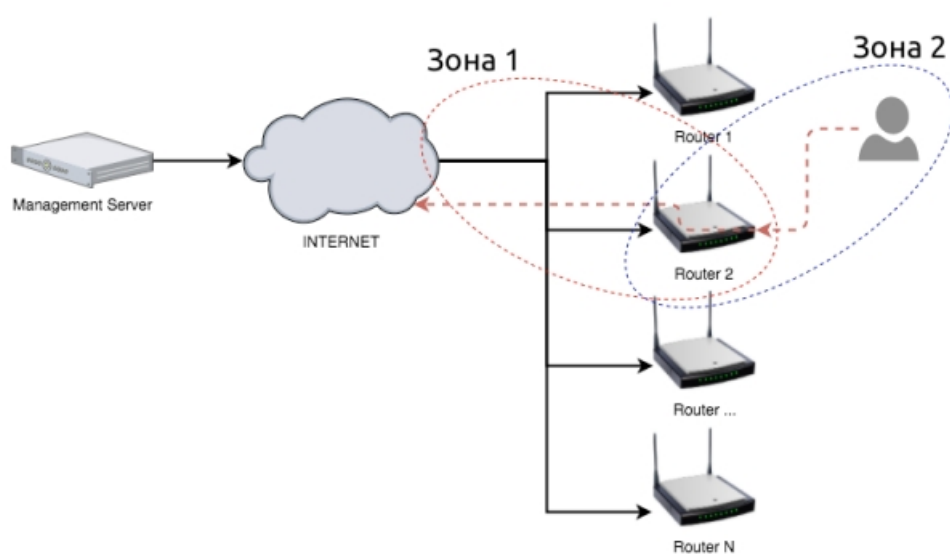
И трафик от "гостя", при всем, при том проходит мимо нашего "управляющего сервера".

Если вы думаете, что такая ситуация совершенно гипотетическая, не встречающаяся в "дикой природе", то нет. Это не только сервис "гостевой Wi-Fi", но и типичная корпоративная сеть для организаций с широкой географией филиальной сети. А в ближайшей перспективе развития систем, например, "интернета вещей", такие сети будут все чаще и чаще появляться, поскольку "вещь" может находиться где угодно, а оператор может быть абсолютно любой - пусть бы и мобильной связи. Или еще запутаннее - LP-WAN с отдельными базовыми станциями в разных городах, странах, континентах.

Важно только обеспечить IP-связанность подобных устройств. И уметь быстро и дешево подключать устройства к любым сетям, любых поставщиков, иначе экономика IoT будет находиться в руках одного поставщика услуг, который экономически логично будет диктовать условия и "ненавязчивый сервис".

Таким образом, мы, как оператор "распределенной сети передачи данных", можем измерять состояние исключительно устройств-элементов нашей сети и управлять только этими устройствами.

Причем, измерять нам придется две зоны:



"Зона 1" - это связанность управляемого устройства вообще с Сетью, а "Зона 2" - попытка измерить связанность "гостя" и управляемого устройства. Оговоримся, что в проекте исследования - это просто "гость Wi-Fi сети", но можно представить на его месте любой другой объект - датчик, исполнительное устройство, корпоративный клиент или даже банальный сетевой принтер.

Таким образом, объект управления мы определили, и попытаемся систематизировать методы мониторинга, но для начала нужно понять, каким образом можно "достучаться" до управляемых устройств через "облако интернет". Выясняется, что эта задача сама по себе не банальна.

Хотя и существует концепция TMN - Telecommunication Management Network, TMN (Система управления сетями операторов электросвязи), которую еще в далеких 199-х годах разработали в Международном Союзе Электросвязи.

Основная идея концепции TMN — обеспечение сетевой структуры для взаимодействия различных типов управляющих устройств и телекоммуникационного оборудования, использующих стандартные протоколы и стеки.

В соответствии с концепцией TMN процесс управления сетью включает в себя следующие функции управления:

1. управление процессом устранения отказов (Fault Management, FM);
2. управление конфигурацией сети (Configuration Management, CM);
3. управление расчётами с пользователями и поставщиками услуг (Accounting Management, AM);
4. контроль производительности сети (Performance Management, PM);
5. обеспечение безопасности работы сети (Security Management, SM).

Эти функции и способы их выполнения достаточно подробно проработаны в рекомендациях ITU серии X.700.

Но, как и многие бюрократические процедурные концепты ITU, были не очень восприняты интернет-сообществом. Хотя некий практически полезный смысл в рекомендациях однозначно присутствует.

Связанные с темой IETF RFC, во многом опираются на рекомендации и соответствуют предложенной модели.

Впрочем, по порядку.

Протоколы управления

Для того чтобы именно управлять (то есть принимать сведения о состоянии элемента системы и выдавать управляющие команды) сетевыми устройствами, вендоры сетевого оборудования придумали самые разнообразнейшие схемы. Некоторые полностью закрыты лицензионными соглашениями и исходным кодом. Некоторые были приняты в качестве стандартов. А некоторые еще в фазе разработки и обсуждений, чтобы возможно стать стандартом.

Попробуем кратко перечислить самые известные и распространенные:

SNMP

Пожалуй, самый известный и распространенный протокол управления сетевыми устройствами на основе TCP/UDP. SNMP - это Simple Network Management Protocol (простой протокол сетевого управления), принятый IETF в качестве утвержденного RFC аж в 1988-ом году. Ну, по крайней мере, [RFC-1065](#), где впервые описана идея и структура протокола, датирован августом 1988-го. Конечно, потом протокол дополняли и улучшали. Уже существует третья версия SNMP, но идеологическая база протокола остается именно из "конца восьмидесятых". По большому счету, вторая версия отличается от первой только появлением прокси-агентов, а третья версия - возможностью шифрования трафика.

Но поскольку протокол имеет "славное прошлое", то почти все существующее сетевое оборудование (и не только - IP-телефоны, принтеры, компьютер-хосты, и даже некоторые аппаратные стойки) имеет поддержку SNMP.

Архитектура систем с поддержкой SNMP - классическая:

- На управляемом устройстве (либо на устройстве, подключенном к интерфейсу управляемого устройства) должен быть запущен SNMP-агент — это специальное программное обеспечение, которое обладает локальным знанием управляющей информации и переводит эту информацию в специфичную для SNMP форму или из неё (медиация данных).
- Система сетевого управления (Network Management System, NMS) — программное обеспечение, взаимодействующее с менеджерами для поддержки комплексной структуры данных, отражающей состояние сети.

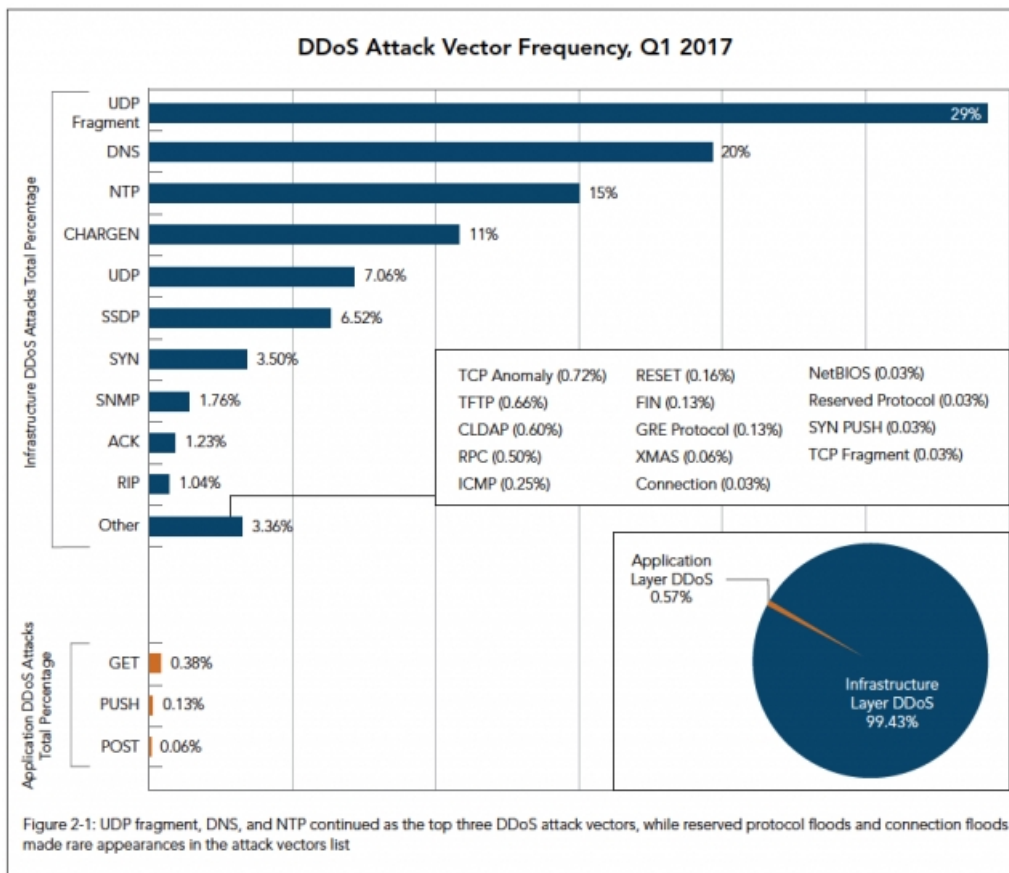
Основное и главное достоинство протокола - это полнота и высокий уровень готовности - IETF признает SNMPv3, определенный в [RFC 3411](#), [RFC 3418](#) (также известный как STD0062) в качестве текущей стандартной версии SNMP. Поддержка протокола действительно очень широка, а его работа очень хорошо изучена большинством сетевых инженеров.

Недостатки же вытекают из изначальной простоты (simple!) протокола - за "один командный проход" в процессе управления выполняется одна функция, что бывает весьма накладно, в случае, если вам необходимо передать достаточно большие объемы управляющей информации. Так, SNMP фактически никогда не используют для передачи таблиц маршрутизации для BGP или IGP.

Рост числа управляемых устройств в сети так же приводит к непропорциональному росту служебного трафика в сети.

Кроме того, SNMP имеет некоторые проблемы с обеспечением безопасности, и даже имеет альтернативную расшифровку в среде специалистов по информационной безопасности - Security Not My Problem. Впрочем, в третьей версии, после ввода в третьей версии криптографической защиты, ситуация сильно улучшилась. Но до сих пор использование SNMP входит в десятку серьезных угроз безопасности.

Вот такой график есть в свежайшем - 1Q2017 - отчете Akamai по направлениям целей DDoS-атак:



Еще один недостаток протокола - плохая визуализация виртуальной модели управляемого устройства. Напомню, что идеология SNMP была разработана еще в "доинтернетную эру", когда сами по себе сетевые приборы были еще достаточно просты, и иерархии MIB-ов вполне хватало для чтения конфигураций человеком. Но со временем сложность устройств росла, а количество реализаций баз управляющей информации (собственно - MIB - Management Information Base) превысило мыслимые пределы понимания. [Открытых MIB на сегодня учтено порядка 7,5 тысяч](#) - это действительно много, поскольку все эти базы было бы неплохо поддерживать на уровне систем менеджмента.

Посему, в 99 случаях из 100, сегодня SNMP используется в качестве протоколов мониторинга и поддерживается большинством систем сетевого мониторинга - Cacti, Zabbix, Nagios и прочие. На [Википедии имеется довольно полный сравнительный список таких систем](#).

CLI

И по всему вышеперечисленному, сетевые администраторы в большинстве случаев предпочитают "старый добрый CLI" - интерфейс командной строки, создавая собственные инструменты управления "на скриптах", что действительно бывает более эффективно для "особых случаев", чем тратить много часов работы на попытки "подружить" NMS и, условно, - "новый BRAS".

Собственно CLI используется всякий раз и для уже функционирующих устройств, когда необходима "тонкая настройка" и/или автоматизация процедур, связанных с конфигурациями большого количества устройств.

В отличие от заготовок MIB или всевозможных графических интерфейсов управления - командная строка иногда просто понятнее и быстрее.

Но требует высокой квалификации инженера, особенно в случае работы на оборудовании, находящемся в "боевой эксплуатации".

CWMP

Однако, с приходом большого количества устройств, находящихся на стороне клиента (CPE - Customer Premises Equipment), стало очевидно, что на каждый DSL-модем, и на каждый Set-Top-Box "командной строкой" не набегаешься. Миллионы, без преувеличения, устройств, находящихся вне физической зоны доступности оператора, тем не менее, требовали обслуживания.

Для чего в середине "нулевых" консорциумом Broadband Forum был разработан специальный протокол управления абонентским оборудованием - CWMP - CPE WAN Management Protocol. Известный также, как TR-069, хотя нужно понимать, что TR-069 - это спецификация, описывающая протокол

CWMP.

В иерархии сетевой модели OSI, этот протокол является прикладным, причем, базируется протокол на базе другой прикладной надстройки над HTTP - SOAP- Simple Object Access Protocol, который, в свою очередь, является "простым описанием информационной модели в нотации XML".

Собственно CWMP в сетевой модели выглядит так:

Физический уровень → Канальный уровень → IPv4/IPv6 → TCP → SSL/TLS → HTTP → SOAP → CWMP.



И становится понятным, что для выполнения функций управления на базе протокола необходимо:

1. Программный агент на управляемом устройстве, поддерживающий CWMP;
2. Управляющее программное обеспечение, которое трансформирует модели в протокол и передает устройствам - ACS - Auto Configuration Servers;
3. Собственно ПО, которое хранит и обрабатывает модели устройств - SCM - Service Configuration Management.

Кроме того, все устройства должны содержать обработчик шифрованного трафика SSL и очень желательно находиться в одной сети, находящейся под управлением оператора. Хотя, это и не обязательно, но прежде всего, каждое устройство должно иметь собственный IP-адрес, иначе как же тогда ACS обнаружит и отконфигурирует устройство?

Преимущества протокола (хотя бы перед тем же SNMP) в том, что сети под управлением TR-069:

- Унифицированное управление сервисами.
- Унифицированное управление по отношению к широкому спектру типов CPE.
- Возможность работы в сетях с использованием NAT.
- Использование XML позволяет относительно легко расширять модель данных.
- Использование TCP в качестве транспортного протокола обеспечивает высокую надежность доставки информации.
- Обеспечивается более высокий, чем в SNMP, уровень информационной безопасности.

Недостатки же прямо вытекают из преимуществ, но главное то, что CWMP не является стандартом, что позволяет производителям CPE вносить в него собственные усовершенствования, существенно усложняющие достижение цели унифицированного управления CPE в сетях провайдеров.

То есть, это достаточно проприетарные реализации, на функционал которых до недавнего времени очень сильно влияли вендоры конкретных решений. Впрочем, в последние год-два появились и реализации на базе Open Source - например, клиенты [Easy CWMP](#) под устройства на OpenWRT или сервер автоконфигурации [GenieACS](#) также доступный в исходных кодах.

NETCONF

О протоколе NETCONF было [подробно рассказано здесь](#).

По сути, NETCONF это то, что называется "лучшие практики" из всех вышеперечисленных протоколов. Но, что интересно, в реальности NETCONF - не протокол, а концепция организации сеанса конфигурации устройства.

Очень упрощенно работу NETCONF можно себе представить как автоматизированный сеанс связи между устройством и сервером по упомянутому уже CLI.

То есть, NETCONF - это представление модели устройства, описанное на специальном мета-языке

YANG (это, по сути, подмножество XML) и очень похожее на идеологию MIB из SNMP-шных реализаций. Но более упорядоченная, стандартизованная и человекочитаемая. Причем, имеется даже отдельный RFC, который описывает перевод MIB в YANG - [RFC6643](#).

И главным преимуществом NETCONF является необязательность наличия агента на стороне управляемого устройства.

Далее, NETCONF описывает процедуру передачи конфигурации и/или данных мониторинга и действий системы в случае коллизий. Но собственно обмен происходит по известным протоколам SSH / SSL или консолью точно так же, как если бы вы вводили команды CLI в ручном режиме.

То есть, по сути, NETCONF - это такое средство автоматизации для CLI (все процессы происходят на стороне серверного П), а именно инициация обмена, выбор изменяемого параметра из модельной базы, проведение операции и нотификация успеха/ошибки всей процедуры.

Достаточно просто и универсально - CLI-интерфейсы есть почти (за очень редким исключением) у каждого сетевого устройства, включая приборы десяти-двадцати летней давности производства.

И если не учитывать того факта, что до сих пор промышленной имплементации протокола NETCONF не существует, то можно сказать, что это идеальный вариант автоматизации управления сетей.

И почему нет имплементаций?

Проблемы адресации

Дело в том, что вышеперечисленные протоколы лишь улучшают модели управления сетевыми устройствами на базе SNMP (напомню, идея 198-х годов), делая взаимодействие системы менеджмента и элементов сети проще, удобнее, быстрее.

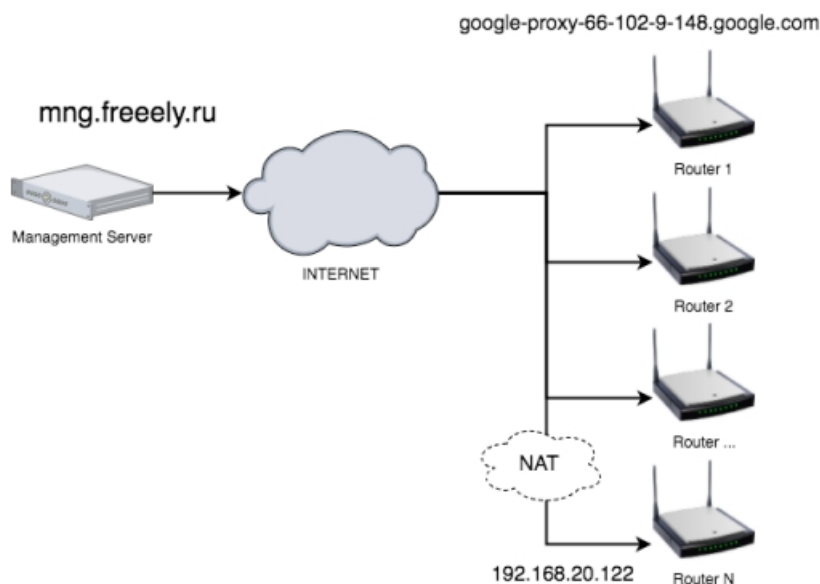
Но при этом не получается никакого нового качества, которое сделало бы организацию-эксплуатанта конкурентоспособнее. По большому счету, администраторам и сетевым инженерам абсолютно все равно, по какому протоколу происходит управление: главное результат в виде управляемой и работающей сети.

Различие между новомодным NETCONF и старинным SNMP просто нивелированы, поскольку результат усилий и там, и там - одинаков.

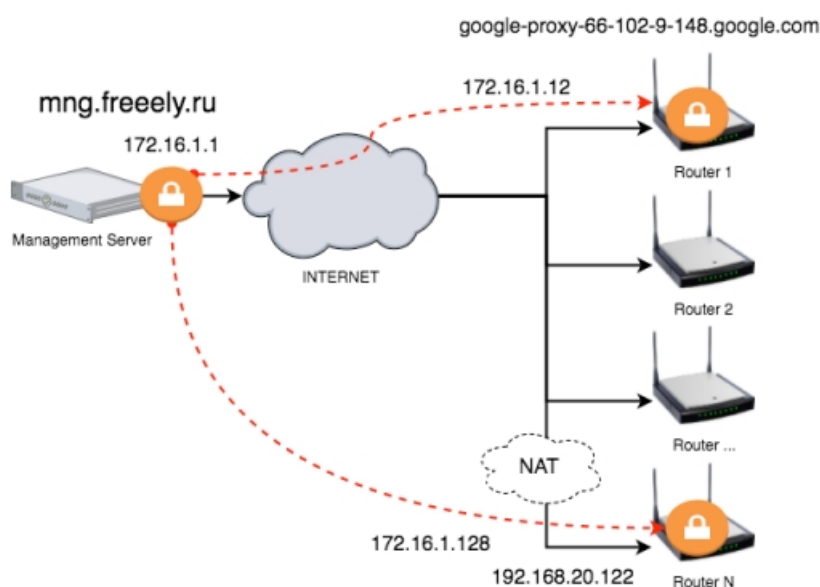
При том, что на имплементацию NETCONF необходимо еще и потратиться, причем и интеллектуально, и материально, то успех концепции довольно сомнителен - старый конь, лучше двух новых понь.

Но хочу напомнить "техническое задание", ради которого тут пишется этот лонг-рид: "Программный комплекс мониторинга и управления распределенными сетями передачи данных для организации систем условного доступа в Интернет". И взглянем на схемы, приведенные выше: в этом случае использование любого описанного протокола проблематично.

Хотя бы потому, что для обращения к устройству, находящемуся "в чужой сети" необходимо знать его фактический IP-адрес, который может быть а) "за NAT'ом" и б) динамически выделяемым. Как-то вот так (все IP-адреса и имена хостов, для примера):



И единственным возможным вариантом организации системы является организация VPN-сетей, где каждый прибор должен быть снабжен клиентом VPN с предустановленным сертификатом и настроенным VPN-сервером на уровне менеджмента:



И в этом случае, используя виртуальные частные каналы и идентификационные сертификаты, наша система может адресовать управляющие команды к устройствам, получивших любой IP-адрес на физическом интерфейсе.

Такая схема включения в настоящее время получила наибольшее распространение для решения задач управления массивом устройств в "наложенных сетях". Причем, использование средств криптозащиты может быть довольно изощренным, а адресация внутри VPN замкнутой внутри такой сети. И даже есть отдельные решения класса "VPN as a Service" - VPNaaS.

Но недостатком решения можно считать необходимость учета и контроля VPN-сертификатов. А установка клиента VPN на устройства нетривиальная задача, если вам необходимо иметь под управлением тысячи единиц таких устройств. Рано или поздно, в такой сети придется заняться

автоматизацией процесса, что потребует слишком много усилий и влечет человеческие ошибки.

RESTCONF

Сетевые инженеры давно задумывались о реализации сценариев управления устройствами класса "подключил и забыл". Или на английском это звучит что-то типа "**Zero Touch Provisioning**". Попытки проприетарных решений появлялись достаточно регулярно. В качестве примера можно привести устройства OpenMesh, которые работают с "облачной системой управления CloudTrax. Или более продвинутое решение - Meraki, купленное на корню компанией Cisco еще в 2012 году за 1,2 миллиарда долларов.

Многие вендоры специализированного оборудования для организации Wi-Fi сетей имеют в портфеле решений что-то очень похожее на "систему конфигурации устройств". Но всеобщего стандарта до сих пор нет.

Точнее, стандарт (RFC точнее) только разрабатывается в недрах IETF.

Только в текущем году появились следующие RFC:

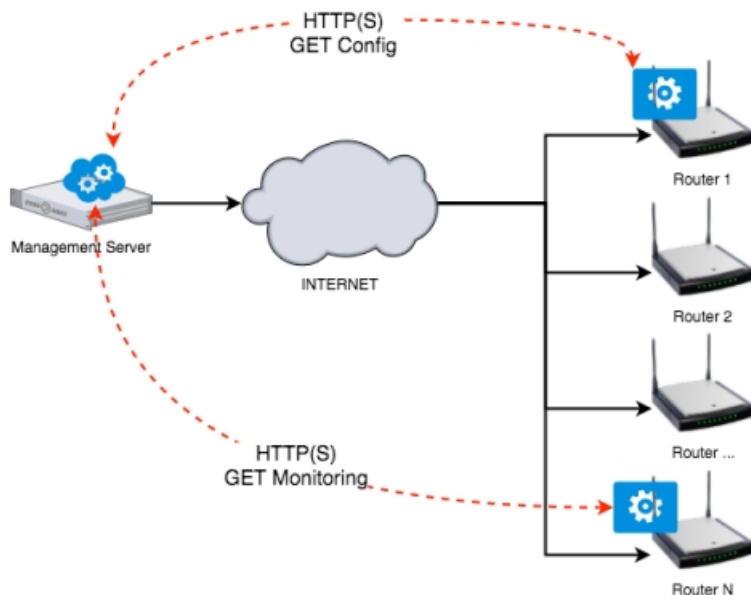
- [RFC 8040 - RESTCONF Protocol](#), утвержденный в январе 2017 года.
- [RFC 8071 - NETCONF Call Home and RESTCONF Call Home](#) (февраль 2017).

Основы RESTCONF

Первый RFC описывает методы подключения к устройствам по HTTP-протоколу, способы аутентификации, структуру сообщений и модели хранения конфигураций устройств, включая мониторинг и обработку коллизий.

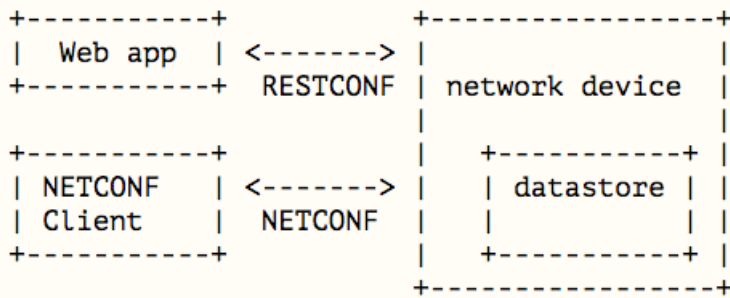
Документ весьма объемный и всеобъемлющий на 137 страниц жесткого технического английского.

Идея RESTCONF заключается в том, чтобы заменить процедуры NETCONF, которые осуществляются по SSH-протоколам на HTTP, чтобы упростить взаимодействие между "сервером управления" и устройством до уровня разработки моделей взаимодействия: сервер и/или устройство посылает запрос -> получает ответ в виде потока данных. Причем, RFC-8040 не настаивает на каком-то точно определенном потоке данных - сообщения могут быть в формате XML или JSON-потоке.



То есть, программирование интерфейсов обмена теперь будет напоминать программирование "банальных html-страничек", что очень серьезно упрощает создание прикладного программного обеспечения на уровне понимания человеком.

При этом RESTCONF совсем не отменяет NETCONF - по задумке авторов RFC они могут работать совершенно параллельно:



Просто для одних и тех же задач можно использовать "или то, или другое" - процедуры соответствуют друг другу:

RESTCONF	NETCONF
OPTIONS	none
HEAD	<get-config>, <get>
GET	<get-config>, <get>
POST	<edit-config> (nc:operation="create")
POST	invoke an RPC operation
PUT	<copy-config> (PUT on datastore)
PUT	<edit-config> (nc:operation="create/replace")
PATCH	<edit-config> (nc:operation depends on PATCH content)
DELETE	<edit-config> (nc:operation="delete")

Попробую еще более упростить для объяснения идеи:

Администратор сети имеет всеобъемлющую базу данных по текущему состоянию всех сетевых устройств. База данных хранится на "сервере управления" и может быть в любом виде, на любом ПО - в зависимости от предпочтений и реализации. Любой MySQL, PostgreSQL или MongoDB - подойдет.

Единственное требование - соответствие модели описания базовым принципам YANG (которых, к слову, уже две версии YANG version 1 [[RFC6020](#)] или YANG version 1.1 [[RFC7950](#)]). Это требование выдвинуто для совместимости описания - ну, чтобы если вы добавляете новый тип устройств в систему, то вам не пришлось бы переписывать систему целиком - просто будет еще одна модель. Примерно, как MIB в старом добром SNMP.

Задумывается, что все модели устройств, методов, обработки ошибок или запросов мониторинга будут публиковаться разработчиками, [например, на Github](#) или на специально созданном "[репозитории моделей](#)", что позволит администраторам просто "накатывать модели" на собственные базы данных и использовать.

В качестве примера приведем описание некоего устройства ACME-System:

```

module acme-system {
  namespace "http://acme.example.com/system";
  prefix "acme";

  organization "ACME Inc.";
  contact "joe@acme.example.com";
  description
    "Модуль устройства ACME для внедрения системы";
}

```

```

revision 2017-01-05 {
    description "Последняя ревизия модуля.";
}

container system {
    leaf host-name {
        type string;
        description "Наименование хоста для системы";
    }

    leaf-list domain-search {
        type string;
        description "Домены для поиска";
    }

    list interface {
        key "name";
        description "Список интерфейсов";
        leaf name {
            type string;
        }
        leaf type {
            type string;
        }
        leaf mtu {
            type int32;
        }
    }
}
}

```

Не думаю, что нужно делать дополнительные объяснения - все достаточно понятно из полей description.

Но вернемся к "простому описанию". Далее администратор просто "работает с соответствующими записями в базе данных". Причем, прошу отметить, что "работа с полями в БД" весьма просто автоматизируется.

Опять-таки пример: в вашей сети (я напомним, это тысячи устройств, которые находятся в разных географических локациях по всему миру) изменяется DNS. Ну, например, появилась необходимость фильтрации контента и вместо DNS, выданного "оператором последней мили", вам нужен свой специфический. В случае если не используете средства автоматизации, вам необходимо "послать команду" смены DNS на каждое устройство и проконтролировать работоспособность каждого узла. На тысячах, напомним.

В случае использования "системы управления", базирующейся на описываемых принципах, - достаточно изменить записи "DNS" в конфигурационных моделях базы данных.

И по логике организация "система управления" должна сама "раскатить" все эти обновления, проверить целостность управляющих команд и протестировать работоспособность изменений.

Собственно, для этого и создавались RFC на NETCON | RESTCONF - все отличие которых, напомним, в протоколах обмена между устройствами и управляющим ПО.

Call Home

Есть еще одна задачка, которую мы упоминали, но в раздел "Основы RESTCONF не вошла". Это - "обход NAT".

Помимо всего прочего в схеме обращения "управляющего сервера к управляемому устройству", есть проблема, связанная с производительностью системы в целом.

Ну, то есть, представьте, что сервер управления расположен где-то "в облаке", а устройств, которым нужно выдать управляющие команды - тысячи. То есть, сервер должен с некоей периодичностью "обходить" все устройства, с целью выдать им команду. Некоторые устройства могут быть недоступны физически - например, просто выключены или выведены из строя. А еще и NAT, напомним.

Особенно пикантна такая ситуация становится для установления соединения с целью получения мониторинговой информации - что там на устройстве происходит? Работает ли оно? Как с загрузкой и вычислительными ресурсами?

То есть, опрос устройств должен быть периодичный. А если мы пытаемся считать метрики, то чем короче период опроса, тем точнее будут расчетные метрики.

Что же делать?

Держать постоянно открытыми SSH-сессии NETCONF? На тысячи и тысячи устройств - это вычислительно затратная процедура, которая рано или поздно займет все ресурсы сервера.

И для решения этой проблемы в IETF был создан еще один RFC: "NETCONF Call Home and RESTCONF Call Home" [[RFC-8071](#)], который, по сути, является дополнением для RFC-8040. Создание отдельного RFC было мотивировано тем, что при полном описании NETCONF и RESTCONF был сделан упор на собственно протоколах. В случае с "Call Home" приводится описание применимости обоих протоколов (неважно какого именно) для частного случая и со следующими мотивациями:

- "Call home" (не знаю, как перевести корректно - "звонок домой" звучит несколько натянуто и порязански) применим как для первоначального развертывания, так и для постоянного управления сетевыми элементами управляемых сетей.
- Любой сетевой элемент при первичном включении может предварительно "позвонить домой", чтобы зарегистрироваться в своей системе управления, тем самым экономя силы и средства системных администраторов.
- Любой сетевой элемент может получить доступ к сети способом, который динамически назначает ему IP-адрес, но не регистрирует назначенный ему IP-адрес службе сопоставления (например, динамический DNS).
- Любой сетевой элемент может быть развернут за брандмауэром, который реализует преобразование сетевых адресов (NAT) для всех внутренних IP-адресов сети.
- Любой элемент сети может быть развернут за брандмауэром, который не позволяет предоставить доступ к внутренней сети.
- Сетевой элемент может быть сконфигурирован в "скрытом режиме" и, следовательно, может не иметь открытых портов для системы управления, что позволяет повысить информационную безопасность эксплуатации сетей в целом.
- Один открытый порт для инициализации управления в центре обработки данных (сервере) в целом предпочтительнее с точки зрения обеспечения информационной безопасности для оператора, чем множество открытых портов на каждом управляемом элементе сети.

То есть, идея заключается в том, чтобы "управляющий сервер" вместо того, чтобы регулярно опрашивать все элементы управляемой сети, просто "слушает обращения" этих элементов и выдает необходимые команды в тот момент, когда устройство обращается к серверу.

И в данном случае как раз протокол RESTCONF более предпочтителен, поскольку работа "управляющего сервера" ничем не будет отличаться от работы обычного web-сервера, который на заданный запрос выдает необходимый в заданной логике ответ. Только вместо привычной веб-странички ответ содержит управляющие команды сетевому элементу. Ну, или сбор данных от средств мониторинга.

Правда, для того, чтобы система заработала, необходим программный агент на каждом сетевом элементе, который хардкодом настроен на собственный сервер управления.

И хочу отметить, что все идеи с "центральной управлением" на базе RESTCONF в IETF еще не закончены.

Продолжение работ

В настоящее время ведется обсуждение следующих драфтов RFC, связанных с развитием идеи. Даты внесения очень свежие, и, надеюсь, что предложения будут приняты с соответствием регламента IETF по отработке стандартов.

- [RESTCONF Changes to Support I2RS Protocol](#)

Дата внесения драфта: 2017-03-29

В предложении описываются процедуры и методы реализации протокола I2RS [[RFC7921](#)] - Interface to the Routing System.

Архитектура I2RS определяет интерфейс I2RS, как "программный интерфейс для передачи состояния в систему Интернет-маршрутизации и из нее". Подход I2RS направлен на использование существующей информации, собранной самим маршрутизатором о топологии сети, ее состоянии, а

также о собственном состоянии и конфигурации. Ограничиваясь интерфейсами уровня системы маршрутизации, I2RS также использует существующие средства преобразования маршрутизационной информации в таблицы передачи FIB (Forwarding Information Base) и таблицы маршрутизации RIB (Routing Information Base).

RESTCONF в данном случае будет использоваться для управления этими таблицами.

- [RESTCONF Client and Server Models](#)

Из названия драфта RFC уже понятно, что речь идет об описании моделей сервера и клиентов в логике использования протокола. Включая конфигурационные настройки клиента и сервера для конфигурирования сетей. Небольшая тавтология. Но это один из самых практичных дарфтов, который позволит, наконец, запустить стандартизацию прикладного ПО - все разработчики ждут "отмашки".

- [Restconf and HTTP Transport for Event Notifications](#)

Еще один ожидаемый драфт по стандартизации транспортных функций реализации RESTCONF'a. Суть заключается в том, что в главном принятом RFC-8040 нет практических описаний процедур "запрос-ответ" на уровне HTTP для различных ситуаций и отработки ошибок. Например, что делать устройству, которое обращается к серверу управления, но если в "управляющей базе данных" выставили, что данное устройство из сети удалено?

Пока описание процедур выглядит так:

```

+-----+
| Subscriber |
| HTTP2 Stream |
| (a) (b) |
+-----+

+-----+
| Publisher |
| HTTP2 Stream |
| (a) (b) |
+-----+

| Restconf POST (RPC:establish-subscription) |
|----->|
| HTTP 200 OK (URI) |
|<-----|
| (7)HTTP POST (URI) (7)
| |----->|
| | HTTP 200 OK |
| |<-----|
| | HTTP Data (event-notif) |
| |<-----|
| Restconf POST (RPC:modify-subscription) |
|----->|
| | HTTP 200 OK |
| |<-----|
| | HTTP Data (subscription-modified) |
| |<-----|
| | HTTP Data (event-notif) |
| |<-----|
| Restconf POST (RPC:delete-subscription) |
|----->|
| | HTTP 200 OK |
| |<-----|
| | HTTP Headers (end of stream) |
| (/7)<-----(/7)

```

Драфт еще неполный и работа над ним ведется.

- [Zero Touch Provisioning for NETCONF or RESTCONF based Management](#)

Еще один практически необходимый драфт RFC, где описывается подключение или отключение устройств от серверов управления (кейс из предыдущего RFC). Каким образом устройство должно получить первичные настройки и описание процедур и методов всего процесса.

- [Accounting in NETCONF and RESTCONF](#)

Естественно, что при массовом использовании систем программной конфигурации и управлением массивами устройств возникнут вопросы по управлению записями каждого устройства, ведение истории взаимодействия устройств и системы управления, ведение логов и учета прав.

Драфт RFC описывает эти процедуры и методы. Например, предлагается, что логирование взаимодействия устройство-сервер будет вестись в следующем формате (пример на YANG):

```
module: ietf-netconf-am
+--ro nam
  +--ro accounting-record* [task-id]
    +--ro task-id      uint32
    +--ro session-id?  nc:session-id-type
    +--ro acct-code    enumeration
    +--ro date-time    yang:date-and-time
    +--ro src-ip       inet:ip-address
    +--ro group        nacm:group-name-type
    +--ro user?        nacm:user-name-type
    +--ro path         nacm:node-instance-identifier
    +--ro action       nacm:access-operations-type
    +--ro rule?        string
    +--ro status?      nacm:action-type
```

Что сделано

Теперь вернемся к цели написания статьи - отчет по проделанной работе в компании "ФРИИ Вай-Фай". Постараемся сделать это сжато и с иллюстрациями.

Но для начала, попытаемся объяснить, зачем это делалось.

Цели, задачи и общая применимость

Изначально, вся система разрабатывалась с целью монетизации публичного Wi-Fi доступа в интернет. Однако опыт эксплуатации таких сетей показал, что:

1. Неуправляемость сетевых элементов ведет к высоким эксплуатационным издержкам и снижению надежности сети. Для контроля качества работы сервиса необходим постоянный мониторинг работы сервиса, как со стороны каждого устройства, так и со стороны серверной части.
2. Стандартное оборудование подвержено вмешательству и атакам по линии информационной безопасности.
3. Сброс устройства в дефолтную конфигурацию приводит к полной неработоспособности системы, а восстановление обходится достаточно дорого, или невозможно, если устройство физически находится в другом городе или даже стране.
4. Любые изменения в архитектуре решения приводит к существенным издержкам.
5. Необходимость ведения истории конфигураций каждого из устройств в сети также влечет издержки. В качестве примера - истечение срока годности SSL-сертификата, установленного на устройствах, приводит к необходимости замены сертификата на всей сети. Автоматизация процесса замены сертификата зачастую просто невозможна.

Кроме того, в процессе разработки и эксплуатации возникли новые требования и запросы от клиентов, возникли новые бизнес-схемы использования технологии:

1. Кроме публичного доступа в интернет, клиенты хотят использовать устройство для внутренних целей, например, для организации частых Wi-Fi сетей. Используемые устройства это позволяют, однако возникает конфликт настроек и требований по безопасности таких решений.
2. Использование систем для целей построения массовых сетей класса HomeSpot, когда устройство абонента может использоваться для организаций частных сетей, когда абонент самостоятельно задает частный шифрованный SSID Wi-Fi сети, задает параметры доступа для нее - пароль и белые списки доверенных mac-адресов. И в то же время, такое устройство может быть использовано для организации операторских сетей публичного Wi-Fi-доступа.
3. Помимо сетей публичного бесплатного Wi-Fi-доступа имеется спрос на решения класса Pay-as-You-Go - организации платного доступа в Сеть через Wi-Fi. Причем существует спрос на решения, которые предлагают оба варианта одновременно - и бесплатный (например, с существенными ограничениями по доступности к ресурсам на основе белых списков или по скорости доступа) и платный (оплата ваучерами или средствами мобильной коммерции) варианты организации

сетей.

4. В корпоративном секторе имеется спрос на системы условного доступа для сотрудников в сочетании с гостевым доступом. Для примера - сотрудникам определенной категории (официанты, продавцы-консультанты и т.п.) доступ в сеть строго лимитирован, при этом гости могут пользоваться любыми интернет-ресурсами. Причем, желательно организовать так, чтобы сотрудник не мог пользоваться гостевыми сервисами.
5. Наступление сервисов "Интернета вещей" также выдвигает особые требования к организации доступа. Например, востребованы решения по организации доступа и контроля за ним для различных устройств, не имеющих интерфейса ввода сетевых параметров. В качестве примера - веб-камеры, которые имеют Wi-Fi интерфейс, датчики открытия, датчики объема и т.п., которые могут переноситься по всей зоне покрытия беспроводной сетью, заменяться и/или добавляться по принципу Zero-Touch исключительно средствами веб-интерфейса администрирования беспроводной сети.

Есть и другие идеи и бизнес-кейсы использования разрабатываемой технологии.

В итоге было принято решение по разработке программного комплекса мониторинга и управления распределенными сетями передачи данных для организации систем условного доступа в Интернет. Основное предназначение комплекса: использование на сетях передачи данных у операторов связи и корпоративных заказчиков в составе комплексных мер по оказанию услуг "условного доступа в интернет".

Разрабатываемый продукт решает следующие задачи заказчиков:

- Получение достоверной и оперативной информации о состоянии устройств и качества оказанных услуг в режиме он-лайн.
- Анализ полученной информации, оперативные сообщения по событиям и отказам устройств, что приводит к улучшению качества услуг и повышению удовлетворенности потребителей услуг "доступ в интернет".
- Значительное сокращение издержек за счет удаленного учета устройств в сети и их конфигураций. Администрирование как отдельных устройств, так и их групп сводится в единый веб-интерфейс, за счет чего один администратор сможет управлять тысячами единиц учета в сети оператора.
- Имплементация бизнес-процессов по учету и управлению пользовательскими устройствами в систему управления.
- Повышение качества оказания услуг доступа (Wi-Fi) за счет регулярного сбора данных о радиобстановке Wi-Fi на управляемых устройствах и оптимизации и оркестровки частотного ресурса Wi-Fi в автоматическом режиме.

При разработке были учтены следующие обстоятельства:

1. Потенциально в управляемой сети могут быть устройства (Wi-Fi маршрутизаторы) различного типа, включая уже действующие сети на базе маршрутизаторов Mikrotik (операционная система RouterOS) и устройства собственной разработки на базе Embedded OS OpenWRT или подобной. Поэтому закладывалась идея использования механизмов, основанных на протоколе NETCONF (для RouterOS), так и RESTCONF для OpenWRT. Поскольку в момент старта проекта соответствующие RFC еще не были утверждены и находились в фазе обсуждения, нам пришлось использовать собственное видение механизмов реализации этих протоколов.
2. Серверных реализаций, полностью использующих механизмы протоколов NETCONF/RESTCONF, до сих пор не существует. Более того, некоторые стандарты реализации функций находятся в процессе разработки и обсуждения в тематических сообществах. Поэтому во многом при разработке мы опирались на собственное видение развития стандартов. В последующем будет необходимо провести работу по приведению системы в соответствие с принятыми стандартами, которые будут объявлены не раньше октября 2017 года. Компания Freeely принимает участие в обсуждении и разработке этих стандартов в рамках форума IETF.

Устройства

В качестве базовых устройств при отработке процедур и механизмов работы протокола NETCONF/RESTCONF были использованы следующие устройства:

Модульный маршрутизатор Dueton Smart Gateway R654G 4G Router:



Особенностью этого устройства является наличие разъема Mini-PCle для установки профессионального 3G или LTE модема, а также вынесенный слот для 1 SIM карты:



На маршрутизаторе установлена операционная система OpenWRT, кастомизированная для работы с системой Freely.

Аппаратные характеристики устройства:

3. Спецификация

3.1 Аппаратные характеристики

Интерфейсы	1x 10/100Mbps WAN/LAN порт
	4x 10/100Mbps LAN порта
	3x USB порта2.0
	1x кнопка RESET
	1x порт RJ11 (FXS)
	1x UART консоль
	1x Mini-PCIe разъем, слот для 1 SIM карты
	1x DC порт (12V/1.5A входное напряжение питания)
Беспроводной стандарт	IEEE 802.11b/g/n
Индикаторы LED	WLAN/System/LAN1/2/3/4/WAN/WWAN
Тип Антенны	2x 2,4GHz/ 5GHz 5dBi RP-SMA M/F 3T3R всенаправленные
Размеры	170mm x 90mm x 28mm (L x W x H)
Процессор	AR9331
Память	64M DDR2
Подключение к мобильной сети	Встроенный 3G/LTE FDD модем Sierra Wireless MC7304 Mini-PCIe

Для данного устройства был разработан специальный агент, который позволяет осуществлять удаленное управление устройством по RESTCONF-подобному протоколу.

Для апробации работы с протоколом NETCONF был использован маршрутизатор Mikrotik RB-951Ui-2HnD. На фото представлен тестовый стенд из шести устройств, которые управляются системой конфигурации:



В качестве базового интерфейса управления устройствами использован стандартный API RouterOS version 3.x.

Для серверной стороны использовалась библиотека [MikroNode](#) для программной платформы [NodeJS](#).

Для организации служебной связи с сервером, на устройствах Mikrotik использовался клиент [OVPN](#).

Базовым устройством в разработке стало следующее устройство:



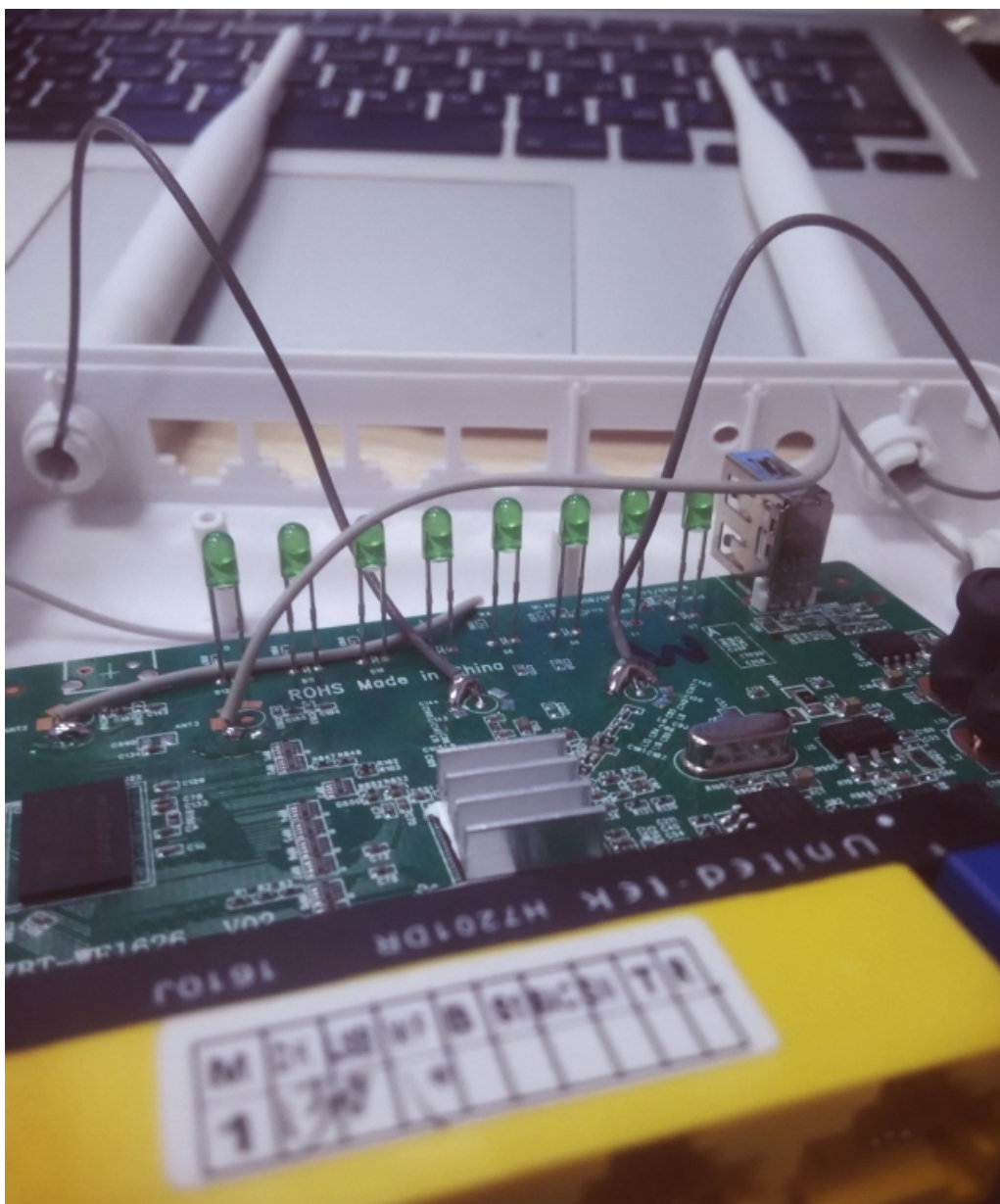
Это инженерный семпл маршрутизатора (без маркировки), производства китайской компании SHENZHEN ELECTRONICS CO.,LTD.

Основные характеристики устройства:

- MT7620N,
- 8MB Flash/64MB RAM,
- OpenWRT Preload,
- 300Mbps Wireless Interface.



Семпл выбирался, в том числе и по экономическим соображениям. Предполагается, что отпускная цена на устройство не превысит 25-30 долларов за устройство.



Основные требования по разработке ПО:

Продукт разрабатывается на основе открытой операционной системы для сетевого оборудования - OpenWrt. Ключевыми особенностями проекта являются:

1. Минимальный веб-интерфейс (см. раздел "Первый запуск и инициализация").
2. Удаленный мониторинг и управление.

Результатом работ является набор исходных текстов и инструменты для сборки готового бинарного образа, программируемого на устройство. Серверная часть управления и мониторинга предоставляется Заказчиком и согласовывается с Исполнителем.

Требования к переносимости

Продукт должен работать на устройствах, построенных на архитектуре MIPS, которые поддерживаются операционной системой OpenWrt. Выбор конкретного устройства осуществляется Исполнителем.

Функциональность

1. Первый запуск и инициализация

При первом включении WAN порта устройства в сеть оператора ПО маршрутизатора пытается получить адрес по DHCP. Если адрес получен, устройство подключается к серверу управления и мониторинга. При невозможности получить адрес, устройство перенаправляет пользователя на страницу конфигурации, где он может задать параметры соединения: статический адрес, L2TP,

PPPoE.

Страница инициализации (первый запуск) устройства - ввод пароля администратора:

Страница конфигурации устройства (первый запуск) - задание параметров доступа к Сети:

[\[Управляется программно: Облачная сеть\]](#)

2. Удаленный мониторинг

Удаленный мониторинг осуществляется путем отправки HTTP запросов на сервер мониторинга. Адрес сервера мониторинга и периодичность отсылки данных задается в настройках.

Параметры мониторинга:

1. Количество подключенных WiFi абонентов.
2. Параметры пинга до заданных ресурсов в сети (время пинга, джиттер).

3. Удаленное управление

Удаленное управление осуществляется через HTTP протокол.

Параметры управления:

1. Управление WiFi:
 - Управление пользовательским SSID (имя точки, пароль, канал).
 - Управление SSID HotSpot Freeely (имя точки, пароль, канал, время сессии абонента).
 - Site Survey - отчет об окружающих точках доступа (имя, канал, сила сигнала).
2. Управление сетевыми параметрами
 - Управление DHCP сервером - диапазон выдаваемых IP адресов.
 - Управление DNS - изменение адреса DNS сервера для клиентов.
3. Удаленная перезагрузка.
4. Удаленная прошивка устройства.
5. Управление логированием.

Серверная часть проекта:

Серверная часть проекта выполнена по функциональному признаку и делится на три части:

1. **Кабинет администратора**, основное назначение которого управление кабинетами партнеров: создание, задания параметров и их изменения. Получение статистики по потребленным ресурсам и услугам каждого партнера.
2. **Кабинет партнера**, который предназначен для:
 - Для управления клиентами - создания, задания параметров и их изменения.
 - Создания тарифных планов.
 - Назначения тарифов клиентов и управления услугами.
 - Получения доступа к аналитике по каждому Клиенту и в целом по сети.

Кабинет Партнера имеет многоязычный интерфейс - русский и английский. В перспективе перевод интерфейса на другие языки.

3. **Кабинет клиента** предназначен для управления услугами сети на стороне клиента (заказчика).

Основной функционал "Кабинета клиента":

1. Просмотр статистики подключения Гостей
 1. Общая статистика
 - Количество подключений новых Гостей (отправлено авторизаций);

- Общее количество подключений;
 - Использование социальных сетей для входа;
 - Пол гостей;
 - Возраст гостей.
- 2. Техническая статистика
 - Доступность устройств;
 - Количество одновременных пользователей он-лайн;
 - Качество сети NQS.
- 3. Статистика сторонних систем веб-аналитики
 1. Яндекс.Метрика (ввод код счетчика - переход на аналитику по ссылке)
 - Ввод ID счетчика в системе по авторизации Freeely
 - Визуализация данных в Кабинете по API
 2. Google Analytics (ввод код счетчика - переход на аналитику по ссылке)
 - Ввод ID счетчика в системе по авторизации Freeely
 - Визуализация данных в Кабинете по API
- 2. Управление Гостями
 1. Список Гостей с возможностью фильтров и сортировок
 - Фильтр по используемой социальной сети;
 - Фильтр по дате регистрации (от и до);
 - Фильтр по дате последнего подключения (от и до);
 - Фильтр по Полу и Возрасту;
 - Фильтр по количеству подключений к сети.
 2. Конкретного Гостя можно:
 - Занести в черный список (выключить), в том числе - в зависимости от условий, например, по расписанию.
 - Перенести в список доверенных (не показывается страница входа или осуществляется перенос в белый список для частной сети).
- 3. Управление устройствами Клиента:
 1. Включить/выключить частный SSID (WPA2-шифрование)
 - Задать собственное имя SSID;
 - Задать пароль для SSID;
 - Задать Wi-Fi-канал;
 - Просмотреть окружение хотспота - Wi-Fi Site Survey;
 - Запретить/разрешить использовать данный SSID определенным mac-адресам.
 2. Включить/выключить публичный SSID
 - Задать собственное имя публичного SSID;
 - Задать время сессий (по умолчанию 1 час);
 - Задать время работы публичного хотспота ("всегда включен" или задать расписание);
 - Задать Wi-Fi-канал;
 - Просмотреть окружение хотспота - Wi-Fi Site Survey.
- 4. Управление политикой подключений Гостей
 1. Включить/выключить социальные сети;
 2. Включить/выключить авторизацию по SMS;
 3. Задать максимальное количество конкретного Гостя подключений в периоде.
- 5. Управление маркетингом
 1. Кастомизация страницы входа (загрузить изображение);
 2. Кастомизация "мини-сайта";
 3. Виджет с API на странице входа.
 4. Триггерный маркетинг
 - Отправка email по событиям
 - Отправка SMS по событиям

Мониторинг

Для целей мониторинга используются следующие функции Системы:

1. Доступность каждого устройства и расчета показателя Uptime используется логирование состояния on-line на сервере OpenVPN.
2. "Качество сети" подключенного устройства осуществляется запуском утилиты Ping на устройстве. Источник мониторинга, частота запуска и прочие параметры задаются в специальной секции конфигурационной модели для каждого устройства. Результаты работы утилиты - количество отправленных пакетов, количество потерянных пакетов, а также время получения пакетов, включая показатели минимальный, максимальный, средний и отклонение mdev - передаются серверу для накопления статистики.
3. Мониторинг работоспособности и нагрузки на радиочасть маршрутизатора осуществляется функцией [SiteSurvey](#) - опрос радиоэфира на предмет доступных (видимых) конкретному устройству точек доступа. Результаты опроса также передаются серверу для ведения статистики и расчета эффективного радиопокрытия путем установки Wi-Fi канала.

Производительность и загрузка радиочасти маршрутизатора осуществляется опросом интерфейса на количество одновременных подключений в текущий момент. Данные по подключенным абонентам также передаются серверу для ведения статистики.

Управление

В системе в настоящее время используется два типа устройств:

1. Устройства на базе OpenWRT обращаются по механизму "Call Home" и протоколу RESTCONF. В ответ устройства получают конфигурационные данные в формате JSON, если конфигурация устройства была изменена после последнего обращения. Если настройки не изменялись, то в ответ устройство получает OK.
2. Устройства на базе RouterOS не имеют механизма самостоятельной инициализации соединения с управляющим сервером, поэтому в случае появления изменений (по событию - нажатие кнопки "применить конфигурацию"), сервер самостоятельно связывается с устройством по протоколу SSH и последовательной передачей команд (API) передает изменения в конфигурации устройства.

Нужно отметить, что пока функционал реализован лишь частично, а собственно такая схема работает довольно нестабильно - соединения в автоматическом режиме успешны в 90% случаев. Требуется доработка механизма обмена данными. Функционал мониторинга устройств на базе RouterOS пока не реализован.

Запрос на сотрудничество

Цель написания данного материала показать, что в Компании "Freeely" ведется довольно серьезная исследовательская работа. Мы внимательно следим за международной работой по направлению стандартизации и описанию систем автоматизации построения сетей и понимаем, как внедрение стандартов может повлиять на развитие телекоммуникационных систем.

Мы добились некоторых результатов в разработке.

Но у нас катастрофически не хватает ресурсов на продолжение исследовательских работ и разработку коммерчески пригодных продуктов.

Если вас заинтересовала наша работа - просим вас обращаться по адресу ceo@freee-wifi.ru для обсуждения деталей сотрудничества.

Спасибо, что дочитали столь объемный материал до конца!