Multicast и все все все ...

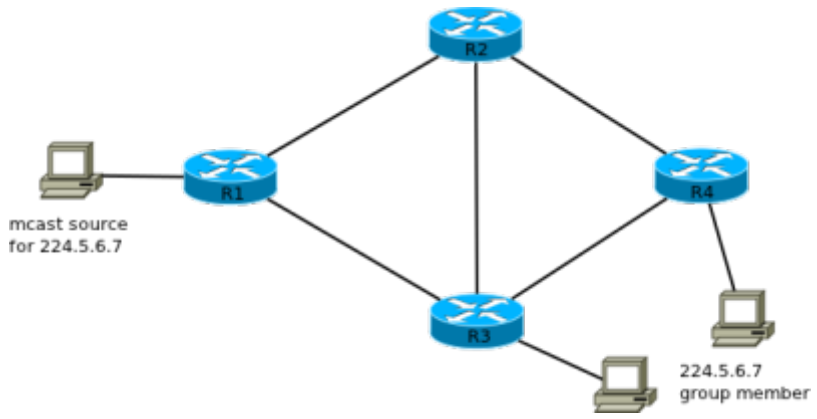
Оригинал статьи

1. Основы передачи multicast.
2. Протокол IGMP.
3. Оптимизация передачи multicast на канальном уровне. IGMP snooping и CGMP.
4. Деревья, применяемые для маршрутизации multicast
5. Протокол PIM-DM
6. Протокол PIM-SM + Cisco AutoRP

Сегодняшняя статья будет посвящена базовым вопросам передачи multicast трафика. Я не стану здесь описывать простых примеров, иллюстрирующих зачем нужен multicast и как multicast позволяет существенно уменьшить трафик сети предприятия или оператора, я полагаю, что читатель представляет себе ответы на эти вопросы. Так же я не стану описывать, что такое классы IP адресов, что такое класс D, чем адреса класса D принципиально отличаются от адресов классов A, B, C. Я полагаю, что читатель по крайней мере понимает особенности адресации групп в IPv4. Поэтому перейдем сразу к делу и начнем обсуждать проблемы, возникающие при маршрутизации multicast.

"Простая" маршрутизация

Положим, у нас есть некоторая сеть, в которой необходимо передавать групповой трафик. Положим, источник multicast передал трафик в некоторую группу 224.5.6.7. Как маршрутизаторы должны обработать такой трафик, чтобы он попал всем членам этой группы независимо от того, где эти члены группы находятся?



Ясно, что можно для начала предложить простое очевидное решение - поступивший трафик маршрутизатор просто должен передать через все свои интерфейсы (очевидно, кроме того, откуда трафик поступил). Проанализируем достоинства и недостатки этого решения и на основании этого анализа сделаем ряд выводов.

Достоинство у такого метода маршрутизации multicast только одно - максимальная простота. Маршрутизатору даже не нужно иметь никаких дополнительных таблиц для маршрутизации группового трафика, просто передаем каждый поступивший пакет через все интерфейсы. Перейдем теперь к анализу недостатков этого подхода, эти недостатки вполне очевидны:

- трафик доставляется даже туда, где он не нужен, т.е. туда, где нет получателей такого трафика. Этот недостаток может быть не столь значителен, если получатели трафика плотно размещены в сети, т.е. почти все являются получателями. Если же получателей multicast в сети не много, т.е. получатели разрежены, то данный недостаток становится серьезным. Также отметим, что даже если в сети почти все узлы хотят получать какой-то multicast, это еще не означает, что эти узлы хотят получать любой multicast. Например, многие узлы в сети оператора получают в данный момент какую-то программу IPTV, но если все узлы начнут получать все программы IPTV, будет не слишком хорошо.

- маршрутизаторы и узлы получают множественные дубликаты пакетов. Это не слишком хорошо, однако в большинстве случае multicast пакеты на прикладном уровне содержат информацию о порядке следования, что в большинстве случаев позволяет приложению, получающему multicast отбрасывать такие дубликаты. Впрочем, пропускной способности сети большое количество дубликатов явно не добавляет.

- Дубликаты пакетов это плохо, но не смертельно, а вот зацикливание пакетов, которое неизбежно будет возникать в топологиях с избыточными связями гораздо менее приятно. На приведенном выше рисунке R1, получив пакет от источника multicast, передает его R2 и R3; R2, получая такой пакет передает его к R3 и R4, а R3, получив пакет от R2, очевидно, в рамках озвученной нами ранее примитивной концепции маршрутизации, передает этот пакет R1. Налицо маршрутная петля, такой пакет будет передаваться в сети (да еще и размножаясь) до тех пор, пока у него не истечет TTL (который маршрутизаторам в multicast пакетах тоже, очевидно, необходимо декрементировать).

Итак, рассмотренный нами выше примитивный алгоритм маршрутизации группового трафика оказывается совершенно не применим, рассмотрим теперь весьма простое улучшение этого алгоритма, которое позволит решить многие описанные выше проблемы.

Концепция RPF - Reverse Path Forwarding

Как видно из приведенного выше примера, маршрутизатор получает много копий одного и того же группового пакета, по одной копии от каждого соседа, и каждая такая копия обрабатывается как независимый пакет, порождая описанные выше проблемы. Как сделать так, чтобы маршрутизатор обрабатывал только одну копию каждого пакета? Очевидное решение: запоминать полученные пакеты и отбрасывать дубликаты. Ясно, что такое «простое» решение никуда не годится, так как создает серьезную вычислительную нагрузку на ресурсы маршрутизатора: нужно либо запоминать все принятые недавно пакеты, либо рассчитывать для каждого принятого пакета контрольную сумму и запоминать эту сумму, а затем каждый вновь поступивший пакет проверять на совпадение с недавно полученными, все это вместе создаст нагрузку как на оперативную память, так и на процессор маршрутизатора. К счастью поставленную задачу можно решить существенно проще, не прибегая к запоминанию недавно обработанных пакетов.

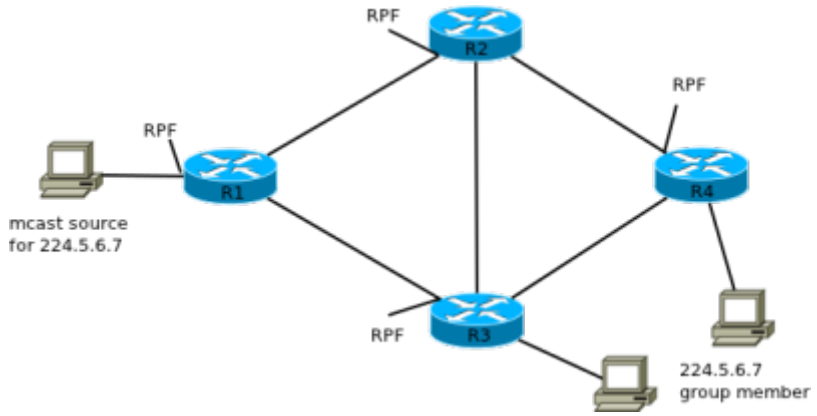
Суть решения состоит в следующем: принимая групповой пакет, маршрутизатор узнает из IP заголовка адрес отправителя (это обычный unicast адрес), на основании этого маршрутизатор может легко определить, как из его интерфейсов ближе к отправителю. Это легко сделать, если у маршрутизатора есть таблица маршрутизации unicast, т.е. обычная таблица маршрутизации: тот интерфейс, через который сам маршрутизатор передавал бы unicast пакеты к отправителю группового пакета и есть самый ближайший интерфейс в сторону отправителя. Такой ближайший к отправителю интерфейс называется RPF интерфейсом, или интерфейсом, прошедшим проверку RPF.

Дальше логика очень проста: если поступивший пакет поступил на маршрутизатор через RPF интерфейс, т.е. через интерфейс, с помощью которого сам маршрутизатор отправлял бы unicast пакет к отправителю, то такой пакет маршрутизатор принимает, если же групповой пакет поступил через любой другой интерфейс, то такой пакет отбрасывается. Почему это работает? Маршрутизатор по-прежнему получает много копий каждого пакета, но принимает всегда только одну копию пакета, ту, которая поступила через RPF интерфейс, при этом маршрутизатору не приходится запоминать недавно обработанные пакеты: в самом отдельно взятом экземпляре пакета находится информация (IP адрес отправителя), которая позволяет принять решение о том, отбрасывать пакет или обрабатывать.

Проанализируем достоинства и недостатки рассмотренной ранее концепции простой маршрутизации группового трафика, которая усилена с помощью технологии RPF. Достоинство по-прежнему одно: простота. Маршрутизатору, как и ранее, не нужна дополнительная таблицы маршрутизации для обработки группового трафика (хотя и нужна «обычная» таблица маршрутизации для работы алгоритма RPF). Теперь недостатки:

- как и ранее трафик доставляется даже туда, где он не нужен, т.е. туда, где нет получателей такого трафика, с этим мы пока ничего не делали
- как и ранее, маршрутизаторы могут получать дубликаты пакетов, но, во-первых дубликатов станет меньше из-за применения RPF, так как не каждый экземпляр пакета, поступивший на маршрутизатор его (маршрутизатор) покидает, а во-вторых теперь у маршрутизатора есть простой механизм для отбрасывания таких дубликатов, т.е. даже если на маршрутизатор через разные интерфейсы поступили копии одного и того же пакета, RPF позволяет отбросить все копии кроме одной.

Наиболее важным плюсом RPF является отсутствие третьего недостатка, который был присущ рассматриваемому простому алгоритму маршрутизации без применения RPF: теперь не происходит зацикливания пакетов в сети: в рассмотренном выше примере маршрутизатор R1 принимает групповой пакет от источника, но даже если затем копию этого же пакета R1 вдруг и получил бы от R3, эта копия будет отброшена и маршрутная петля не будет иметь места.



Можно ли использовать описанную выше простую маршрутизацию группового трафика с поддержкой RPF в реальных условиях? С одной стороны на данном этапе такой подход выглядит достаточно приемлемым, по крайней мере он не порождает глубоких проблем благодаря RPF, с другой стороны остается первая проблема: трафик по-прежнему затапливает сеть, т.е. доставляется даже туда, где нет получателей, ясно, что RPF никак не может решить эту проблему. Если в сети очень мало группового трафика, в идеальном случае один источник, одна группа вещания и почти все узлы постоянно хотят получать этот трафик, такой подход мог бы считаться приемлемым. Но в реальных условиях, когда в сети потенциально много

источников, точно много групп вещания, а получатели распределены в сети более или менее разреженно, передавать весь групповой трафик всех групп на все узлы сети конечно же не приемлемо. Как же поступать?

Ясно, что следующим шагом будет ограничить доставку группового трафика только тем узлам, которые в нем нуждаются, не затапливая трафиком всех групп всю сеть. Для этого, очевидно, необходимо отойти от изложенного выше принципа «простой маршрутизации», вместо этого необходимо сформировать у маршрутизатора таблицу маршрутизации milticast, с помощью которой трафик будет передаваться только туда, где он необходим. Рассмотрим вопрос о том, как должна выглядеть такая таблица маршрутизации.

Таблица multicast маршрутизации

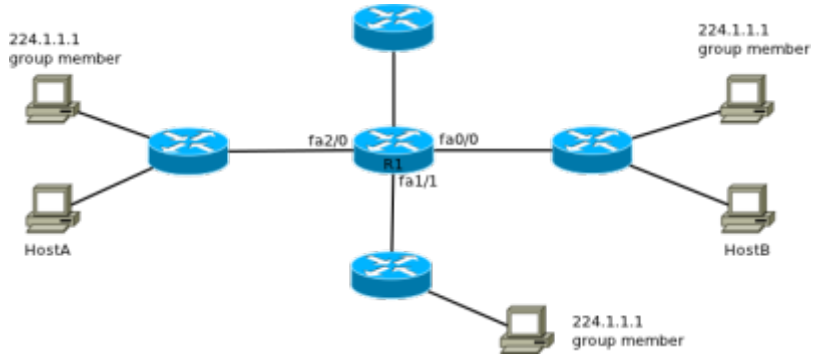
При обработке unicast трафика, маршрутизаторы используют таблицы маршрутизации, типичная структура которых предполагает, что каждому префиксу (т.е. номеру сети и маске) ставится в соответствие next hop (а еще, возможно, выходной интерфейс, метрика, время жизни записи в таблице и др., но это сейчас не столь важно). Иными словами, когда пакет поступает на маршрутизатор, в таблице маршрутизации сдержится инструкция о том, куда этот пакет отправить, причем ответ на вопрос «куда пакет отправить» не зависит ни от чего, кроме как от поля destination ip в заголовке поступившего пакета (да, я знаю про load balancing, policy routing и другие вещи, которые могут, помимо destination ip, влиять на то «куда отправить поступивший пакет» но это сейчас нас скорее отвлечет, поэтому не будем обращать на это внимания). И, что крайне важно, при обработке unicast маршрутизатор, получая экземпляр пакета, всегда отправляет через выходной интерфейс только один экземпляр маршрутизированного пакета.

Важнейший вопрос: выполняются ли эти же два правила для обработки multicast, т.е.:

- зависит ли от чего либо еще, кроме multicast destination ip ответ на вопрос «куда передать поступивший на маршрутизатор multicast пакет?
- получая multicast, передает ли маршрутизатор дальше только один экземпляр этого пакета?

Легко видеть, что оба эти правила не применимы для обработки multicast пакетов маршрутизатором. Пример ниже иллюстрирует, что маршрутизатор R1, получая multicast пакет, посланный HostA (или HostB), должен передать его через два интерфейса, в противном случае некоторые потенциальные получатели не смогут получить этот пакет, иными словами, получив один экземпляр пакета, маршрутизатор должен отправить через разные интерфейсы два экземпляра этого пакета.

Так же ясно, что решение о том, куда отправить пакет с multicast адресом получателя, маршрутизатор не может принимать только на основании адреса получателя.



Пусть сперва пакет с неким адресом получателя 224.1.1.1 отправил узел HostA. Тогда маршрутизатор R1 должен отправить такой пакет через интерфейсы fa0/1 и fa1/1. Пусть теперь пакет на 224.1.1.1 отправил HostB, в этом случае маршрутизатор должен отправить пакеты через интерфейсы fa1/1 и fa2/0. Очевидно, что при принятии решения о том, как маршрутизировать multicast пакет, необходимо принимать во внимание не только адрес получателя, но и адрес отправителя пакета.

Как же должна выглядеть строка в таблице multicast маршрутизации с учетом описанных выше особенностей? Во-первых, вместо префикса сети назначения в такой строке должно быть два параметра: multicast адрес получателя и unicast адрес отправителя, обычно это записывается в формате (S, G), где S - unicast адрес источника, а G - групповой адрес получателя, в нашем примере (192.168.0.95, 224.1.1.1). Во-вторых в качестве цели в строке должен присутствовать не next hop, а список интерфейсов, через которые нужно передавать полученный пакет, в нашем примере строка в таблице маршрутизации будет выглядеть так:

```
(192.168.0.95, 224.1.1.1) - fa1/1, fa2/0
```

Как видим, структура таблицы маршрутизации группового трафика достаточно отличается от аналогичной структуры для unicast трафика.

Итак, теперь, когда мы сформулировали, что для эффективной маршрутизации multicast необходима специальная таблица маршрутизации, зададимся последним на сегодня вопросом: как сформировать такую таблицу маршрутизации?

Введение в multicast маршрутизацию

Вспомним принципы формирования unicast таблиц маршрутизации. Для того, чтобы все маршрутизаторы в автономной системе имели актуальные таблицы маршрутизации, между этими маршрутизаторами работает некий IGP (OSPF, RIP, EIGRP, ISIS), с помощью которого маршрутизаторы обмениваются информацией об имеющихся маршрутах. Очевидно, для маршрутизации multicast нам понадобится некоторый протокол маршрутизации, выполняющий аналогичные IGP функции.

Кратко вспомним, как принципиально работает любой IGP, не взирая на различия между ними. Когда мы говорим, что IGP позволяет маршрутизаторам обмениваться маршрутами, мы предполагаем, что им есть чем обмениваться, т.е. у маршрутизаторов изначально есть некие маршруты, различные у различных маршрутизаторов в AS, которыми они, собственно, и обмениваются. Конечно же такими изначальными маршрутами являются маршруты в подключенные сети (directly connected network). Фактически перед началом работы IGP сведения о доступности КАЖДОЙ сети в AS известные по крайней мере одному

маршрутизатору, конечная цель работы IGP в том, чтобы сведения о доступности КАЖДОЙ сети в AS стали известны КАЖДОМУ маршрутизатору.

Имеет ли место аналогичная ситуация с multicast? Увы, НЕТ. Маршрутизаторы multicast не знают (по крайней мере пока, без специальных мер) кто за его портами решил получать multicast трафик какой группы, фактически для обмена сведениями о multicast маршрутах по аналогии с unicast маршрутами у multicast маршрутизаторов нет (пока) исходных данных, аналогичных сведениям о directly connected сетях в unicast.

Соответственно, задача об эффективной маршрутизации multicast должна быть разделена на две части:

- Необходим механизм, с помощью которого маршрутизаторы будут узнавать от узлов, подключенных к ним, к каким группам эти узлы подключены в данный момент, причем, в отличие от directly connected сетей в unicast, эта информация меняется постоянно. Для решения этой задачи служит протокол IGMP (Internet Group Management Protocol), который будет рассмотрен в следующей статье.
- После того, как маршрутизаторы снабжены механизмом «узнавания» «directly connected» групп, необходимо реализовать собственно протокол маршрутизации, т.е. механизм обмена данными о подключенных группах таким образом, чтобы все, кто необходимо получал необходимый multicast, а все остальные не получали лишнего трафика. Для решения этой задачи сегодня применяют семейство протоколов PIM (Protocol-Independed Multicast), которое будет рассмотрено в одной из следующих статей.

Подведем краткие итоги:

- Применение концепции RPF позволяет организовать»простую« (без таблиц маршрутизации и протоколов маршрутизации) маршрутизацию multicast трафика в небольшой сети, в которой мало (лучше один) источников трафика, мало групп вещания (лучше одна) и много получателей трафика (лучше все узлы). Ясно, что ситуация, при которой такая «простая» маршрутизация эффективна совершенно гипотетическая, однако сама концепция RPF нами будет в дальнейшем активно использоваться в более эффективных, нежели «простая» маршрутизация, схемах работы. Кроме того рассмотрение недостатков «простой» маршрутизации полезно для понимания сложностей, возникающих при маршрутизации multicast, а также для понимания того, почему в дальнейшем задача о маршрутизации multicast решается так, а не иначе.
- Таблицы маршрутизации multicast становятся необходимы, как только мы хотим уйти от «простой» маршрутизации и доставлять multicast трафик только тем, кто в нем нуждается. Структура multicast таблицы маршрутизации отличается от соответствующей таблицы unicast маршрутизации.
- Для решения задачи об организации протокола multicast маршрутизации необходимо сперва решить задачу об информировании маршрутизаторов о том, какие группы подключены к маршрутизатору в каждый момент времени.

To be continued …

Теперь ясно, что прежде чем говорить о протоколах multicast маршрутизации, необходимо обсудить протокол IGMP, с помощью которого маршрутизаторы получают начальные знания о том, за какими их портами присутствуют члены каких групп вещания. Именно протоколу IGMP

будет посвящена следующая статья этого цикла.

Протокол IGMP

Оригинал статьи

Данная статья является второй, посвященной multicast маршрутизации, с первой статьей можно познакомиться по ссылке. В первой статье мы остановились на том, что для организации протокола маршрутизации, который позволил бы маршрутизаторам обмениваться сведениями о подключенных multicast группах, сперва необходим механизм, с помощью которого маршрутизаторы сами узнавали бы о том, какие multicast группы подключены к их интерфейсам. Эта задача в ipv4 решается с помощью протокола IGMP, которому и посвящена данная статья.

Сразу скажу, что из всего запланированного цикла статей это - самая простая. Протокол IGMP сам по себе не сложен, просто и понятно описан в rfc, кроме того в Сети достаточно много статей, описывающих его функциональность, поэтому я постараюсь сделать обзор этого протокола очень подробным и проиллюстрировать его примерами трафика, который читатель не найдет ни в rfc, ни, как правило, в других статьях.

Итак, положим, что в автономной системе существует совокупность маршрутизаторов, к котором подключены узлы, поддерживающие получение multicast. С помощью IGMP каждый из маршрутизаторов должен получить актуальные в каждый момент времени сведения о том, члены каких multicast групп присутствуют за каждым интерфейсом маршрутизатора, впоследствии эта информация будет использована как основа для обмена данными в рамках протокола динамической маршрутизации multicast трафика. Рассматривая сегодня IGMP, мы не будем интересоваться тем, как узлы подключены к маршрутизаторам, т.е. не будем интересоваться L2 топологией и, что самое главное, не будем интересоваться оптимизацией передачи multicast в L2 топологиях. Пока будем полагать, что узлы подключены к маршрутизаторам с помощью L2 коммутаторов, которые обрабатывают multicast простым flooding, как и broadcast, т.е. передают полученный multicast кадр на все порты, исключая порт, с которого кадр поступил. Такой подход позволит нам быстро разобраться с поставленной задачей, т.е. понять, как маршрутизаторы поддерживают актуальные сведения о подключенных членах групп. Что же касается вопросов оптимизации L2 инфраструктуры, этому будет посвящена следующая статья.

Для решения поставленной задачи, узлы и порт маршрутизатора в каждой L2 сети обмениваются пакетами протокола IGMP, к непосредственному рассмотрению которого мы и переходим.

Протокол IGMP предполагает асимметричный обмен данными между маршрутизаторами и узлами, т.е. маршрутизаторы условно говоря «спрашивают» о членстве в группах, а узлы условно говоря «отвечают» о таком членстве. IGMP пакеты инкапсулируются непосредственно в IP пакеты, поле Protocol в IP заголовке устанавливается в 0x02, IGMP заголовок в таком пакете следует за IP заголовком.

В rfc988 описывается нулевая версия IGMP, которая не имеет сегодня никакой практической ценности/области применения и полностью устарела. Поэтому мы начнем рассмотрение в версии 1, которая описана в rfc1112. Эта версия сегодня считается устаревшей, однако может применяться старыми устройствами и, в отличие от нулевой версии, совместима с

современными версиями IGMP.

В rfc1112 описывается заголовок IGMPv1, приведенный ниже, никаких дополнительных данных за этим заголовком не должно следовать.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Type  |    Unused     |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Group Address                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Поле Version в rfc1112 полагается равным 1.

Поле Type указывает на тип IGMP сообщения, таких типов в IGMPv1 всего два, и пара 4-х битовых полей Version + Type принимает следующие значения

Type = 0x11 - Host Membership Query. Данное сообщение посылают только маршрутизаторы, с помощью этого сообщения маршрутизаторы спрашивают узлы за интерфейсом о том, членами каких групп являются узлы.

Type = 0x12 - Host Membership Report. Данное сообщение посылают узлы, с помощью этого сообщения узлы сообщают маршрутизатору о том, членами каких групп узлы являются. Сообщение данного типа узлы могут посылать как в ответ на Query, полученный от маршрутизатора, так и по собственной инициативе, желая уведомить маршрутизатор о членстве в новой группе/группах.

Поле Unused не используется, в отправляемых пакетах оно должно быть равным 0x00, в принятых пакетах это поле необходимо игнорировать.

Поле Checksum используется для защиты от всего IGMP сообщения (т.е. заголовка длиной 8 байт), для расчета контрольной суммы пакета само поле Checksum полагают равным нулю. Очевидно, что полученные пакеты с неправильной контрольной суммой необходимо игнорировать.

Поле Group Address в пакетах типа Query должно быть установлено равным нулю в отправляемых Query и игнорироваться в принимаемых Query. В пакетах типа Report данное поле предназначено для указания адреса группы, членом которой узел является, информируя тем самым маршрутизатор о наличии за интерфейсом члена группы с указанным Group Address.

Теперь рассмотрим логику обмена данными между интерфейсом маршрутизатора и узлами с помощью описанных выше сообщений Query и Report. Маршрутизаторы, поддерживающие multicast периодически рассылают через свои интерфейсы Query для того, чтобы узнать, члены каких групп присутствуют за тем или иным портом. Такой Query рассылается с destination IP 224.0.0.1 (это специальный адрес, называемый «все поддерживающие multicast хосты», принимать пакеты, отправленные по такому адресу, как следует из названия должны все устройства, поддерживающие работу с multicast), такой пакет имеет ip ttl = 1. В общем случае каждый узел обязан в ответ на такой Query отправить пакет Report для КАЖДОЙ группы, членом которой он является (за исключением группы 224.0.0.1, членство в этой группе ожидается по умолчанию), причем Report о членстве к конкретной группе посылается на адрес этой группы, т.е. поле destination ip в заголовке Report пакета совпадает с номером группы, о

членстве в которой отправлен Report. Как видно из формата заголовка IGMP, один пакет Report сообщает о членстве только в одной группе (с помощью поля Group Address), таким образом узел, получив Query посылает столько пакетов типа Report, в скольких группах узел состоит.

В таком поведении узла есть два недостатка:

- В ответ на Query может немедленно прийти очень много Report, так как каждый узел шлет отдельный пакет для информирования о членстве в каждой группе, что при большом числе узлов и большом числе групп может перегрузить сеть.
- Маршрутизатору не обязательно знать, сколько членов некоторой группы есть за интерфейсом. Независимо от того, есть ли за интерфейсом один член группы 225.1.2.3 или членов такой группы 1000, маршрутизатор в любом случае должен перенаправлять трафик этой группы через этот интерфейс.

Для уменьшения трафика пакетов Report применяются следующие подходы, оптимизирующие описанную выше логику поведения узла при получении Query:

Первый описанный выше недостаток потенциально приведет к очень большому количеству Report, отправляемых немедленно после передачи маршрутизатором Query, для устранения этого узел, получая Query, должен для каждой группы, членом которой он является, запустить специальный таймер (report delay timer), и только по истечению данного таймера отправить соответствующий Report о членстве в данной группе. Для каждой группы узел выбирает случайный таймер в диапазоне от 0 до 10 секунд, таким образом весь описанный выше трафик пакетов типа Report в локальной сети, в которую маршрутизатор отправил Query равномерно распределяется в 10-ти секундном интервале, устраняя тем самым потенциальную возможность перегрузки сети пакетами типа Report.

Для устранение второго недостатка узел поступает таким образом: если узел еще не отправил Report о членстве в некоторой группе (так как еще не истек report delay timer для этой группы), а другой узел сети отправил Report о членстве в этой же группе, то узел останавливает report delay timer для данной группы и отказывается от отправки Report в данную группу. Это приводит к тому, что сколько бы ни было членов некоторой группы за интерфейсом маршрутизатора, он получает только один Report о членстве в каждой конкретной группе. Очевидно, это возможно потому, что Report, как было указано выше, узлы шлют на адрес группы, о членстве в которой сообщают, позволяя тем самым и другим узлам, которые являются членами данной группы, слышать эти Report и подавлять свои.

Описанные выше оптимизации позволяют с одной стороны уменьшить общее количество Report, генерируемых в ответ на отправленный маршрутизатором в сеть Query, а с другой стороны «распределить» оставшиеся Report в 10-ти секундном интервале, устранив тем самым описанные выше недостатки.

Маршрутизаторам необходимо отправлять свои Query периодически для того, чтобы всегда иметь актуальную информацию о том, члены каких групп присутствуют за интерфейсами маршрутизатора, rfc1112 рекомендует делать это не чаще одного раза в 60 секунд. Однако при старте маршрутизатора рекомендует отправлять несколько Query с небольшими интервалами, чтобы маршрутизатор мог как можно оперативнее получить необходимую информацию.

Узлам, помимо описанных выше ответов на Query, рекомендуется при присоединении к группе немедленно, не дожидаясь Query, отправлять Report о членстве в новой группе на тот случай, если узел первым в данной сети является членом этой группы, кроме того этот пакет

рекомендуется повторять через краткое время 1-2 раза (на случай потери/повреждения пакета).

Теперь проанализирует недостатки IGMPv1, приведшие, в конечном счете к появлению новой версии протокола. Можно выделить три основных недостатка IGMPv1, непосредственно следующих из описанных выше принципов работы протокола:

- Покидая группу, узлы никак не информируют об этом маршрутизатор. Если узел перестает быть членом группы, и так получилось, что данный узел был последним членом данной группы, то маршрутизатор, не получая никакой информации о том, что подписчиков данной группы у него больше за этим интерфейсом нет, продолжает отправлять уже никому не нужный multicast трафик группы через данный интерфейс. Каким вообще образом в IGMPv1 маршрутизаторы узнают о том, что подписчиков той или иной группы за интерфейсом больше нет? Перестав получать Report о членстве в данной группе в ответ на Query, т.е. неявно и с заметной задержкой.
- Невозможность для маршрутизатора явно запросить о членстве в определенной группе. Так как единственный запрос, который может послать маршрутизатор это Query, т.е. запрос о членстве во всех группах, то запросить о членстве в определенной группе можно только послав Query, получить множество ответов и отфильтровать эти ответы на предмет поиска ответа об интересующей группе. Эта проблема перекликается с предыдущей: если маршрутизатор в ответ на Query не получил ответа о членстве в группе, в которой ранее были члены, то, вместо того, чтобы послать явный запрос о членстве в указанной группе чтобы уточнить, правда ли в ней больше нет членов, маршрутизатор вынужден снова слать Query, в ответ на который поступят ответы о членстве во всех группах.
- Отсутствие диалога между multicast маршрутизаторами. Положим, два или более маршрутизатора подключены к одной локальной сети. Тогда каждый маршрутизатор будет независимо от других отправлять Query в сеть, порождая тем самым в ответ поток Report, хотя, очевидно, учитывая изложенное выше, достаточно, чтобы это делал только один из них. Кроме того каждый маршрутизатор потенциально будет передавать multicast всех групп, для которых в локальной сети есть подписчики в эту сеть, порождая избыточный трафик.

Все эти проблемы в конечном счете привели к разработке протокола IGMPv2, описанного в rfc2236, к рассмотрению которого мы переходим. Заголовок IGMPv2 похож на заголовок IGMPv1 и совместим с ним:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     | Max Resp Time |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Group Address                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Рассмотрим отличия в заголовке:

- Вместо двух 4-х битовых полей Version и Type теперь есть одно 8-и битовое поле Type
- Поле Unused заменено на поле Max Respond Time, которое имеет следующий смысл: будучи установлено в пакетах типа Query, оно указывает, в каком интервале от 0 до Max Respond Time нужно выбрать случайный report delay timer перед отправкой Report в ответ

на данный Query. Значение в поле Max Respond Time указывается в десятых долях секунды, таким образом перед отправкой Report необходимо сделать случайную паузу в интервале от 0 до (Max Respond Time) /10 секунд. С помощью этого поля маршрутизатор может регулировать задержку, с которой он получает ответ на посланный Query, очевидно, что чем меньше Max Respond Time, тем большая нагрузка порождается ответами на соответствующий Query.

Рассмотрим типы IGMPv2 пакетов и сравним их с рассмотренными ранее типами IGMPv1 пакетов:

- Type = 0x11 - Memebership Query. Данный тип пакет делится на 2 подтипа:

 # General Query. Такой пакет отправляется маршрутизаторами по адресу 224.0.0.1, поле Group Address заполняется нулями и такой пакет является полным аналогом Host Membership Query пакете в IGMPv1. Более того, обратите внимание, что значение поля Type в этом пакете тождественно значению пары полей Version + Type в Host Membership Query пакете в IGMPv1. Фактически, независимо от того, какую версию IGMP (1 или 2) поддерживает узел, получивший такой пакет, узел интерпретирует такой пакет как предложение сообщить о членстве во всех группах, которым узел принадлежит.

 # Group Specific Query. Такой пакет маршрутизатор отправляет по адресу некоторой группы (а не на 224.0.0.1), и в поле Group Address указывается тот же адрес группы, что и в заголовке IP пакета. Данный пакет позволяет маршрутизатору задать вопрос о наличии членов в одной конкретной группе, чей адрес и указывается в поле Group Address, что в свою очередь позволяет маршрутизатору понять, остались ли в данной группе активные члены. Данный пакет является частью процедуры, связанной с выходом узлов из группы. В IGMPv1, как мы помним, явный выход из группы вообще не предусмотрен, в IGMPv2 эта проблема решена двумя сообщениями: рассмотренным Group Specific Query и пакетом Leave Group, который будет рассмотрен ниже. Аналога данного пакета нет в IGMPv1, однако IGMPv1 узел, получая такой пакет, как было сказано выше, должен игнорировать поле Group Address, а это значит, что узел рассматривает такой пакет как обычный Host Membership Query, просто отправленный по групповому адресу, а не по адресу 224.0.0.1 (а значит получат такой пакет только члены группы, которой пакет отправлен на L3, а не все узлы.

- **Type = 0x16 - Membership Report v2**. Данный тип пакета полностью аналогичен применяемому в IGMPv1 пакету Host Membership Report, с помощью одного такого пакета узел, поддерживающий IGMPv2 сообщает маршрутизатору о членстве в одной multicast группе, разумеется, узел должен в ответ на General Query послать столько пакетов данного типа, в сколько группах узел состоит. Отличие этого пакета от IGMPv1 Host Membership Report только в том, что применяется иное значение поля Type, что позволяет маршрутизатору понять, что он имеет дело с IGMPv2 клиентом, а не с IGMPv1 клиентом.
- **Type = 0x17 - Leave Group**. Данный пакет отправляется узлами по адресу 224.0.0.2 (все multicast маршрутизаторы), в поле Group Address узел указывает, какую группу он покидает. Это сообщение решает важную проблему IGMPv1 - проблему молчаливого выхода из группы. Однако вспомним: маршрутизатору важно знать, есть ли за его интерфейсом хотя бы один член некоторой группы, поэтому выход из группы только одного члена еще не означает, что маршрутизатор должен перестать передавать трафик этой группы через интерфейс, получив Leave Group, маршрутизатор должен выяснить, остались ли его интерфейсом другие члены данной группы, или же данный член группы был последним. Именно для это и используют рассмотренный выше Group Specific Query: получив Leave Group пакет, маршрутизатор посылает для этой группы Group Specific

Query, пытаясь понять, остались ли в этой группе другие члены, и если маршрутизатор получает Report в ответ на такой пакет, он продолжает посылать трафик для указанной группы через интерфейс. Учитывая, что Report в ответ на General Query шлют не все члены группы, а, обычно, только один узел у которого первым истек report delay timer, применяется следующая оптимизация: если узел слал свой Report о членстве в указанной группе в ответ на последний General Query, то он, выходя из группы, должен отправить Leave Group пакет. Если же узел не отправлял Report о членстве в данной группе в ответ на последний Query, это значит, что в группе гарантировано есть и другие члены данной группы, в таком случае узел может покидать группу молча.

- **Type = 0x12 - Membership Report v1**. Данный тип пакета непосредственно является Host Membership Report в IGMPv1 и поддерживается в IGMPv2 для совместимости с узлами, которые поддерживают IGMPv1. С помощью одного такого пакета узел, поддерживающий только IGMPv1 сообщает маршрутизатору о своем членстве в одной группе, разумеется, такой узел посылает столько пакетов типа Membership Report v1, в сколько группах такой узел состоит. Таким образом узлы, поддерживающие IGMPv2 сообщают о своем членстве в группах с помощью серии пакетов с типом 0x16, а поддерживающие только IGMPv1 - с помощью пакетов с типом 0x12.

Итак, подведем итоги по рассмотренному формату пакета IGMPv2. В IGMPv2 добавилось по сути три новых типа пакета:

- Report о членстве версии 2 (наряду с сохранением Report о членстве версии 1) - для того, чтобы маршрутизатор знал, какие версии клиентов есть за его интерфейсом
- Leave Group пакет, позволяющий узла явно сообщить о выходе из группы
- Group Specific Query пакет, позволяющий маршрутизаторам проверить, остались ли еще члены группы, для которой получен Leave Group пакет.

Новые пакеты и привнесенный ими новый функционал решают две из трех описанных ранее проблем IGMPv1, однако остается еще одна проблема: отсутствие диалога между multicast маршрутизаторами, иными словами несколько multicast маршрутизаторов в одной сети независимо отправляют свои Query, порождая тем самым лишний IGMP трафик. Для устранения этой проблемы в IGMPv2 вводятся понятия Querior и Non-Querior. В одной физической сети типично только один multicast маршрутизатор должен выполнять функции Querior, все остальные должны быть Non-Querior. Именно Querior должен отправлять в физическую сеть все Query (как General Query так и Group Specific Query), анализировать же поступившие в ответ Report должны все multicast маршрутизаторы. Так как Non-Querier, получая Leave Group пакет не имеет право послать в ответ на него необходимый Group Specific Query, Non-Querior должен игнорировать Leave Group пакеты, но при этом Non-Querior может получать информацию о группах, в которых нет членов, анализируя Group Specific Query, посланные Querior и Reports, поступившие в ответ. Как происходят выборы Querior? Каждый маршрутизатор, начинающий работать с IGMP через интерфейс должен полагать себя Querior и должен начать отправлять Query в физическую сеть. При получении Query от другого маршрутизатора с меньшим IP адресом отправителя, маршрутизатор должен немедленно стать Non-Querior. Если Non-Querier в течение определенного интервала не получает Query, он должен стать Querier и начать сам отправлять Query.

Теперь поговорим о совместимости IGMPv1 и IGMPv2. Сперва рассмотрим ситуацию, когда в сети, где в основном используется IGMPv2, появляется IGMPv1 маршрутизатор. Сперва рассмотрим, как должен вести себя IGMPv2 узел, если Query пакеты в сети отправляет IGMPv1 маршрутизатор.

Узлу не сложно понять, что Query послан IGMPv1 маршрутизатором - в таких пакетах поле Max Response Time = 0x00 потому, что в IGMPv данное поле не используется и должно быть обнулено. В этом случае узел обязан:

- Полагать, что Querier в сети является IGMPv1 маршрутизатор не на основании последнего полученного Query, а на основании истечения специального таймера, обновляемого всякий раз, когда получен IGMPv1 Query, т.е. даже получение одного или нескольких IGMPv2 Query после IGMPv1 Query еще не означает, что Querier стал IGMPv2 маршрутизатор. Это станет ясно ниже, когда мы рассмотрим как должны вести себя IGMPv2 маршрутизаторы если в сети есть IGMPv1 маршрутизатор.
- полагать поле Max Response Time = 100 (т.е. выдерживать случайную задержку перед ответом на Query в интервале 0 - 10 секунд) .
- не отвечать на такие Query с помощью Membership Report v2, так как такой тип пакета игнорируется IGMPv1 маршрутизатором.
- отвечать на Query с помощью Membership Report v1
- не отправлять Leave Group сообщений, так как они игнорируются маршрутизатором

Теперь рассмотрим как должен вести себя IGMPv2 маршрутизатор в сети, где есть хотя бы один IGMPv1 маршрутизатор. В этом случае rfc2236 требует, чтобы маршрутизаторы, поддерживающие IGMPv2 были административно настроены использовать в данной сети IGMPv1.

Как обеспечивается совместимость в том случае, когда в инфраструктуре IGMPv2 присутствуют IGMPv1 узлы? Если IGMPv2 узел получает Membership Report v1/Membership Report v2 о членстве в группе после General Query, он должен подавлять свой собственный Membership Report v2. Если маршрутизатор получил Membership Report v1 для некоторой группы, он должен:

- запустить для этой группы таймер, фиксирующий наличие в этой группе IGMPv1 узла
- обновлять этот таймер всякий раз при получении Membership Report v1
- игнорировать Group Leave сообщения для этой группы

Теперь, когда мы разобрались с протоколами IGMPv1 и IGMPv2 рассмотрим несколько примеров трафика этих протоколов. Для начала подключим к коммутатору два маршрутизатора Cisco, один из них будет выступать в качестве маршрутизатора multicast, а второй будет выступать в качестве клиента (разумеется, в качестве второго устройства можно подключить и PC, запустив там, ПО, которое будет пытаться стать членом multicast группы. Я в своих примерах буду в качестве узлов использовать маршрутизаторы Cisco, а в качестве multicast приложения буду использовать обычный пинг из IOS) .

Сперва настроим оба устройства на использование IGMPv1. Начальный конфиг «маршрутизатора» выглядит так:

```
R1(config)# ip multicast-routing ! *
R1(config)# interface FastEthernet1/0
R1(config-if)# ip address 192.168.0.1 255.255.255.0
R1(config-if)# ip pim dense-mode ! **
R1(config-if)# ip igmp version 1 ! ***
R1(config-if)# end
```

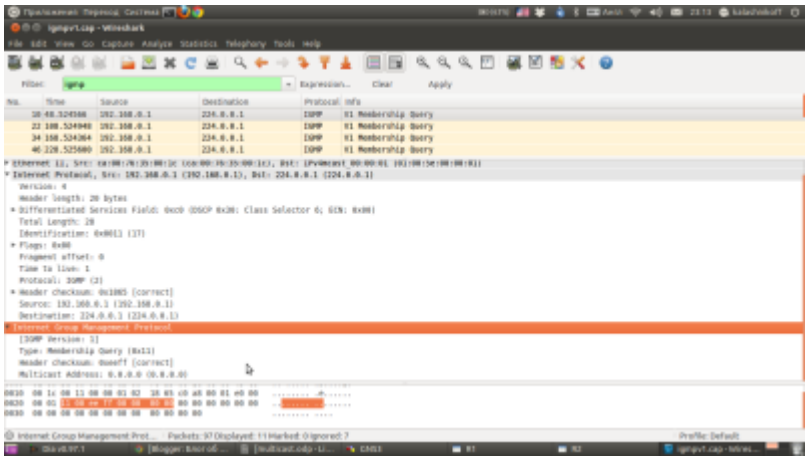
* Эта команда включает поддержку маршрутизации multicast в IOS **Эта команда включает конкретный протокол маршрутизации multicast. Для нас сейчас совершенно не важно,**

даже какой именно протокол маршрутизации milticast включать, в данный момент мы только лишь изучаем IGMP, так что включим первый попавшийся протокол просто для того, что маршрутизатор начал заниматься отправкой IGMP Query и анализом поступающий IGMP Reports. * Данная команда, как несложно догадаться, административно требует форсированного использования IGMPv1 на данном интерфейсе.

Начальный конфиг «узла» еще проще, там даже не нужно включать поддержку multicast маршрутизации, соответственно и не нужно включать и протокол маршрутизации. Просто настроим интерфейс «узла» и снова таки форсируем использование IGMPv1:

```
R2(config)# interface FastEthernet1/0
R2(config-if)# ip address 192.168.0.100 255.255.255.0
R2(config-if)# ip igmp version 1
R2(config-if)# end
```

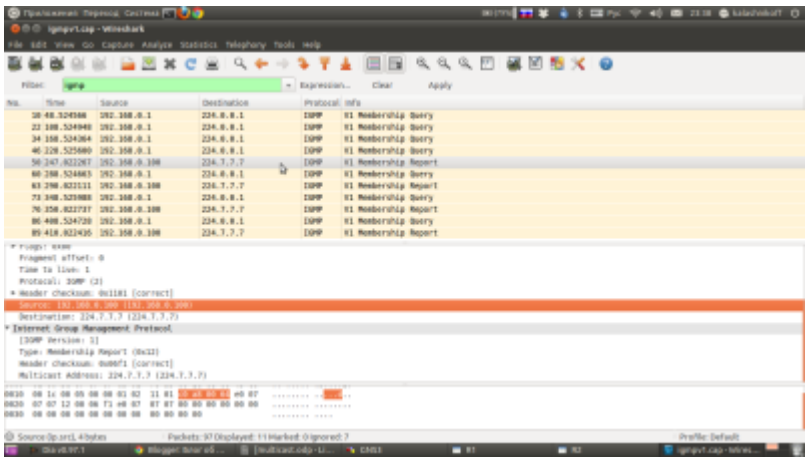
Теперь проанализируем трафик в сети между двумя устройствами. Очевидно, пока там будут присутствовать только IGMPv1 Query, периодически посылаемые R1:



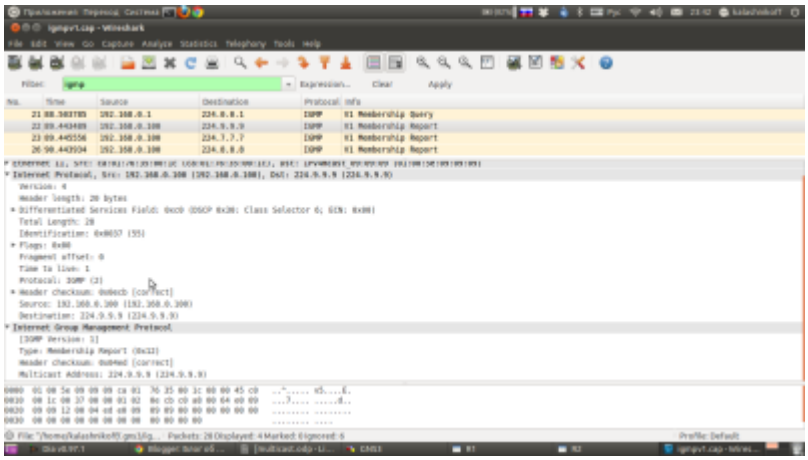
Теперь сделаем интерфейс «узла» членом некоторой multicast группы с помощью команды

```
R2(config-if)# ip igmp join-group 224.7.7.7
```

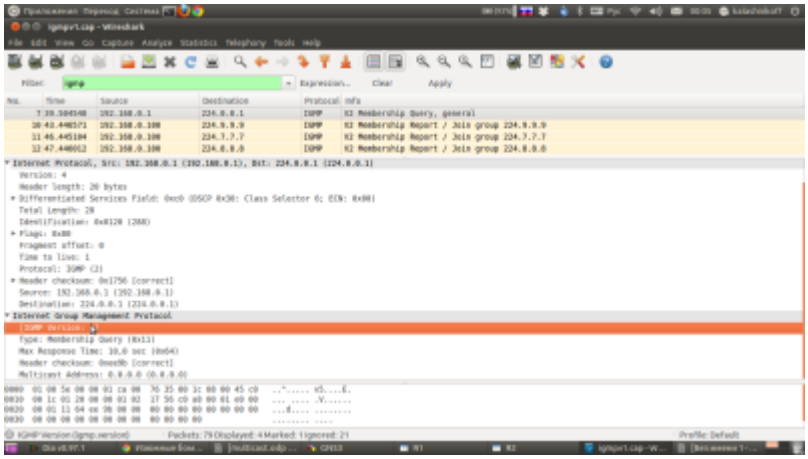
Продолжим анализ трафика. Теперь мы увидим во-первых Report, который посылает «узел» не дожидаясь очередного Query (для немедленного получения multicast трафика), а во-вторых регулярные Report, посылаемые «узлом» в ответ на каждый Query «маршрутизатора».

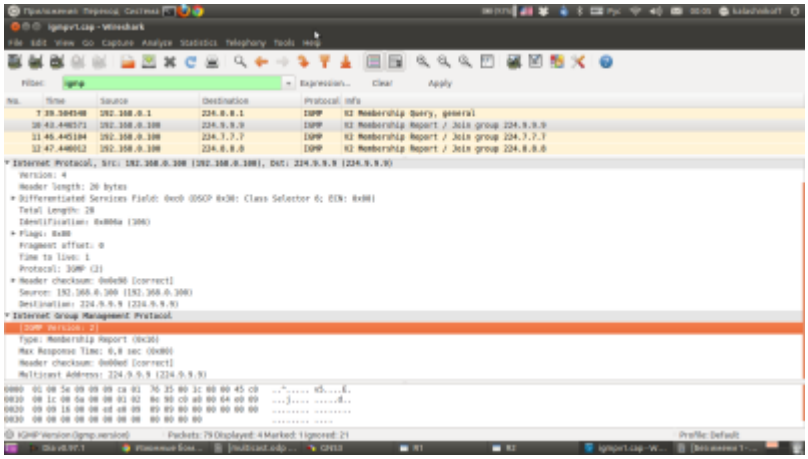


Ну и, завершая с IGMPv1, сделаем наш «узел» членом нескольких групп и убедимся, что отчет о членстве в каждой отправляется в отдельном Report

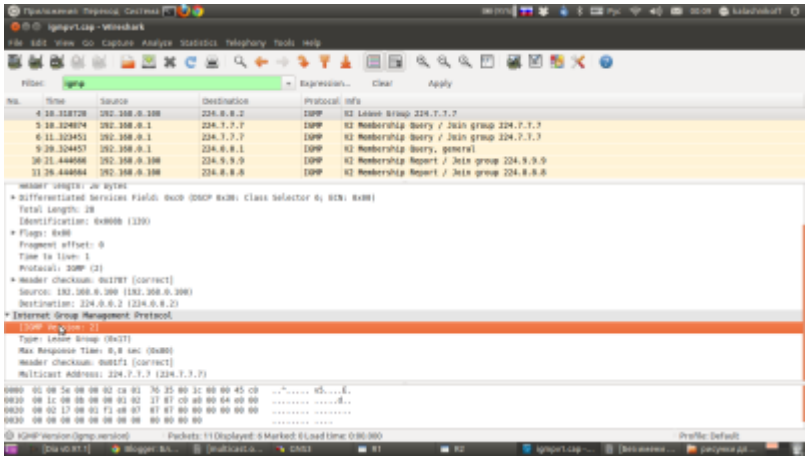


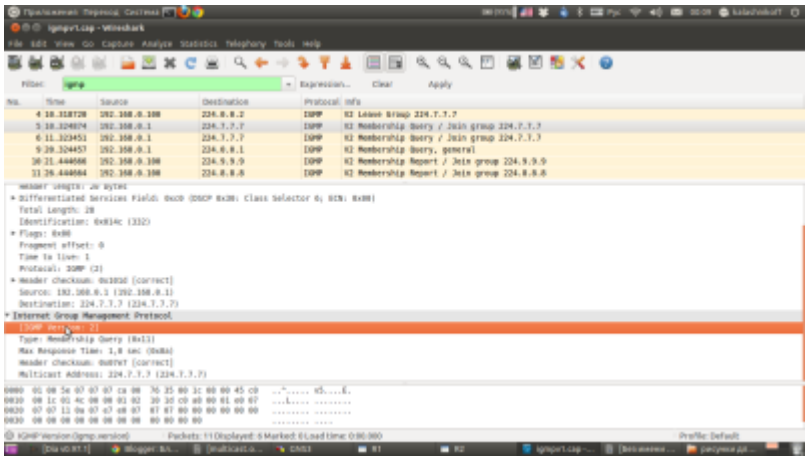
Теперь форсируем использование IGMPv2 и рассмотрим формат General Query и v2 Memebership Report:





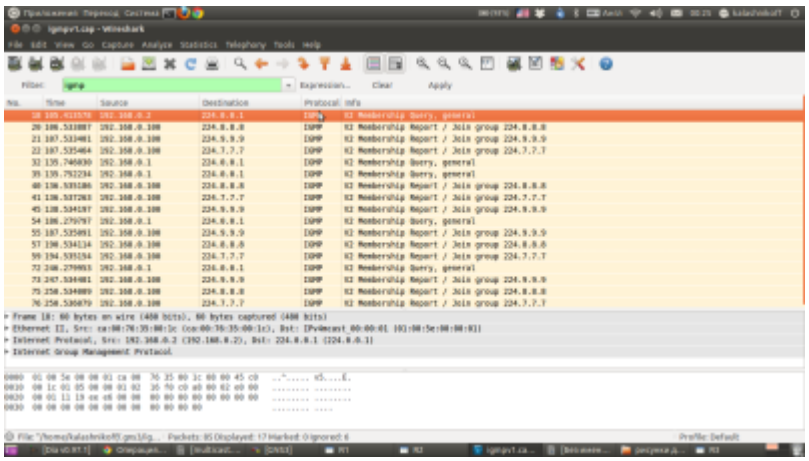
Наконец, рассмотрим процесс выхода «узла» из членов группы 224.7.7.7





На первом скриншоте мы видим пакет Group Leave, после чего маршрутизатор дважды отправляет Group Specific Query в группу 224.7.7.7 чтобы убедиться, что в ней больше нет членов (их нет так как на эти пакеты не поступило ответа), после чего мы видим обычный периодический Query и два (а не три) Report от «узла».

Наконец последний пример иллюстрирует выборы Querier в IGMPv2 окружении. На иллюстрации видно, что General Query посылал маршрутизатор с адресом интерфейса 192.168.0.2, однако на 135 секунде наблюдений в сети появился еще один маршрутизатор, который немедленно послал два General Query, после чего, так как IP адрес этого маршрутизатора меньше, первый маршрутизатор посчитал, что перестал быть Querier и перестал посылать в сеть General Query.



Помимо рассмотренных IGMPv1 и IGMPv2 существует и третья версия протокола - IGMPv3. В этой версии добавлена поддержка управления multicast подписками в зависимости от

источника трафика, иными словами клиент с помощью IGMPv3 может не просто запросить трафик некоторой группы, но и указать, от каких источников клиент хочет получать трафик. Эта версия сегодня применяется, хотя и реже, чем IGMPv2. В данный момент мы не будем рассматривать IGMPv3, возможно в одной и следующих статей мы вернемся к этому вопросу.

SSM

Всё, что мы описывали до сих пор — это **ASM — Any Source Multicast**. Клиентам безразлично, кто является источником трафика для группы — главное, что они его получают. Как вы помните в сообщении IGMPv2 Report запрашивается просто подключение к группе. **SSM — Source Specific Multicast** — альтернативный подход. В этом случае клиенты при подключении указывают группу и источник. Что же это даёт? Ни много ни мало: возможность полностью избавиться от RP. LHR сразу знает адрес источника — нет необходимости слать Join на RP, маршрутизатор может сразу же отправить Join (S, G) в направлении источника и построить SPT. Таким образом мы избавляемся от

- Элемент ненумерованного спискаПоиска RP (протоколы Bootstrap и Auto-RP),
- Регистрации источника на RP (а это лишнее время, двойное использование полосы пропускания и туннелирование)
- Переключения на SPT.

Поскольку нет RP, то нет и RPT, соответственно ни на одном маршрутизаторе уже не будет записей (*, G) — только (S, G). Ещё одна проблема, которая решается с помощью SSM — наличие нескольких источников. В ASM рекомендуется, чтобы адрес мультикастовой группы был уникален и только один источник вещал на него, поскольку в дереве RPT несколько потоков сольются, а клиент, получая два потока от разных источников, вероятно, не сможет их разобрать. В SSM трафик от различных источников распространяется независимо, каждый по своему дереву SPT, и это уже становится не проблемой, а преимуществом — несколько серверов могут вещать одновременно. Если вдруг клиент начал фиксировать потери от основного источника, он может переключиться на резервный, даже не перезапрашивая его — он и так получал два потока.

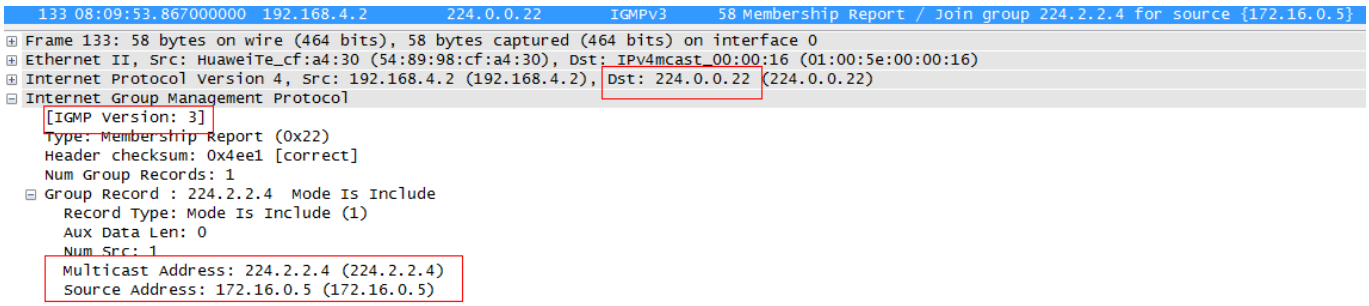
Кроме того, возможный вектор атаки в сети с активированной мультикастовой маршрутизацией — подключение злоумышленником своего источника и генерирование большого объёма мультикастового трафика, который перегрузит сеть. В SSM такое практически исключено.

Для SSM выделен специальный диапазон IP-адресов: 232.0.0.0/8. На маршрутизаторах для поддержки SSM включается режим PIM SSM.

```
Router(config)# ip pim ssm
```

IGMPv3 и MLDv2 поддерживают SSM в чистом виде. При их использовании клиент может

- Запрашивать подключение к просто группе, без указания источников. То есть работает как типичный ASM.
- Запрашивать подключение к группе с определённым источником. Источников можно указать несколько — до каждого из них будет построено дерево.
- Запрашивать подключение к группе и указать список источников, от которых клиент не хотел бы получать трафик



IGMPv1/v2, MLDv1 не поддерживают SSM, но имеет место такое понятие, как **SSM Mapping**. На ближайшем к клиенту маршрутизаторе (LHR) каждой группе ставится в соответствие адрес источника (или несколько). Поэтому если в сети есть клиенты, не поддерживающие IGMPv3/MLDv2, для них также будет построено SPT, а не RPT, благодаря тому, что адрес источника всё равно известен. SSM Mapping может быть реализован как статической настройкой на LHR, так и посредством обращения к DNS-серверу.

Проблема SSM в том, что клиенты должны заранее знать адреса источников — никакой сигнализацией они им не сообщаются. Поэтому SSM хорош в тех ситуациях, когда в сети определённый набор источников, их адреса заведомо известны и не будут меняться. А клиентские терминалы или приложения жёстко привязаны к ним. Иными словами IPTV — весьма пригодная среда для внедрения SSM. Это хорошо описывает концепцию **One-to-Many** — один источник, много получателей.