EBU – Tech 3346

End-to-End IP Network Measurement for Broadcast Applications

EisStream Software package description

Geneva
May 2011

EBU – TECH 3346

End-to-End IP Network Measurement for Broadcast Applications

EisStream Software package description

Внимание!
Данный перевод **НЕ** претендует на аутентичность
и может содержать отдельные неточности.
Оригинал документа на сайте https://tech.ebu.ch

Сквозное измерение IP-сетей для вещательных приложений

Описание пакета программного обеспечения EisStream

Женева

Май 2011

Система обозначений

Настоящий документ содержит как **нормативный,** так и **информативный** текст.

Весь текст является нормативным, кроме Введения, разделов, отмеченных как «информативные», или отдельных параграфов, начинающихся с «Примечания».

Нормативный текст описывает обязательные или непреложные элементы. Он содержит ключевые слова «должен», «следует» или «можно», определяемые следующим образом:

«Должен» или «не должен»: (обязательно)	Указывает требования, которые нужно строго соблюдать и от которых не допускается отклонений для соответствия документу.
«Следует» или «не следует»: (рекомендуется)	Указывает, что один из нескольких вариантов рекомендуется как особенно подходящий, не упоминая и не исключая других. ИЛИ что определенный ход действий предпочтителен, но не обязателен. ИЛИ что (в отрицательной форме) определенный вариант или ход действий не рекомендуется, но не запрещается.
«Можно» или «можно не»: (опционально)	Указывает ход действий, допустимый в рамках документа.

По умолчанию означает обязательные (в фразах, содержащих «должен») или рекомендуемые (в фразах, содержащих «следует») предустановки, которые могут быть опционально изменены пользователем или иметь другие опции в продвинутых приложениях. Обязательные установки по умолчанию должны поддерживаться. Поддержка рекомендуемых установок предпочтительна, но не обязательна.

Информативный текст потенциально полезен для пользователя, но не обязателен и может быть исключен, изменен или дополнен, не влияя на нормативный текст. Информативный текст не содержит ключевых слов соответствия.

Совместимая реализация включает все обязательные условия («должен») и все рекомендуемые условия («следует») в случае их реализации. Совместимая реализация не требует реализации опциональных условий («можно»).

Содержание

Сквозное измерение IP-сетей для вещательных приложений
Описание пакета программного обеспечения EisStream

Комитет EBU	*Первый выпуск*	*Переработка*	*Переиздание*
EC-M	2011		

Ключевые слова: Интернет-протокол, Сетевое управление, Вещание, MIB, EisStream

1. Введение

В последние годы члены EBU все чаще используют IP сети для контрибуции аудио и видео в реальном времени. Известно. что хотя IP сети недороги и дают больше гибкости по сравнению с сетями с коммутацией каналов, они имеют больше задержек и гораздо больше джиттера, а толерантность вещателей к этим переменным гораздо ниже, чем к обычному деловому IT трафику.

Для решения проблем членов с использованием IP EBU создал две группы, ECN-ACIP (Audio contribution over IP) и ECN-VCIP (Video contribution over IP) с задачей разработки рекомендуемых норм практики[1].

Также была понятна высокая потребность в инструментах, позволяющих вещателям правильно измерять и управлять своими IP сетями в соответствии со множеством критичных ко времени приложений. Для этой цели была создана группа ECN-IPM (IP measurement). Отношения между этими группами показаны на следующем рисунке.

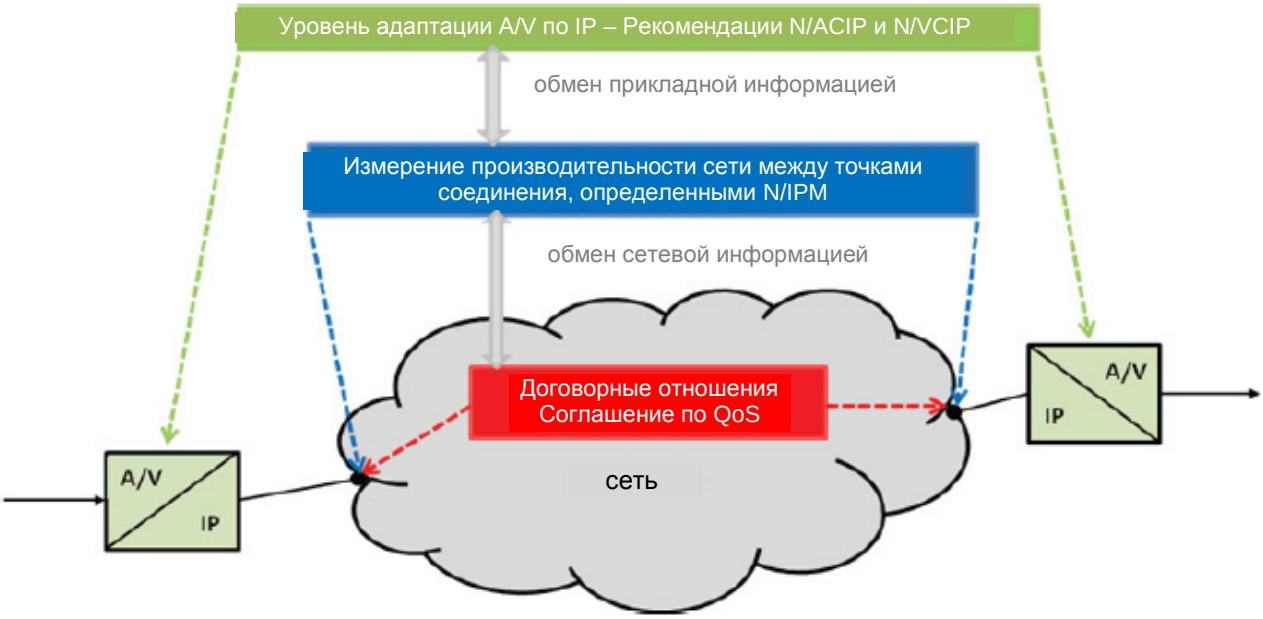

Рис. 1: Связь между ECN группами ACIP, VCIP и IPM

Задачи группы ECN-IPM были следующими:

- Определить классификацию качеству услуг для достижения требуемого качества передачи A/V для вещания.
- Стандартизировать сетевой обмен информацией между членами EBU и провайдерами сетей.
- Предложить метод сбора информации о сквозной производительности в целях управления.

Выполняя эти задачи, группа ECN-IPM определила ряд параметров, важных для вещателей при использовании IP сетей для передачи аудио и видео, и разработала механизм программного обеспечения, чтобы опробовать сеть для обнаружения устройств и топологии, отслеживания физического тракта для сквозной связи и многоадресных потоков, с потенциалом для многоуровневого мониторинга потоков в мультивендорной сети с параметрами, полностью зависящими от медиа.

[1] ECN-ACIP и ECN-VCIP ранее известны как N/ACIP и N/VCIP соответственно.

Указанные параметры включают как сетевой, так и прикладной уровень (для видео и аудио). SNMP используется для сбора информации о статусе сетевых устройств, например, скорости передачи, коэффициенте ошибок, используемом кодеке и статусе многоадресных потоков.

Чтобы гарантировать восстановление всех параметров из всего множества IP оборудования разных производителей, группа разработала MIB (Management Information Base). Хотя за долгие годы было опубликовано много файлов MIB, особенно со стороны сети, было проделано очень мало работы по стандартизации MIB файлов A/V кодека. Поэтому группа EBU ECN-IPM предложила новый стандарт на основе IEC 62379 (Common Control Interface for Networked Audio and Video Systems) для решения этой проблемы.

Технические публикации EBU были сделаны группой ECN-IPM:

EBU Tech 3345 определяет параметры и новую информацию MIB.

Настоящий документ, **EBU Tech 3346**, является описанием программного механизма **EisStream**[2]. Пакет программного обеспечения написан на Java и обеспечивает отслеживание физического тракта для IP трафика с использованием SNMP.

2. Характеристики и функциональные возможности

Новый стандарт мониторинга, определенный в EBU Tech 3345, адресован уникальной проблеме, вставшей перед вещателями в мониторинге сквозного медиа трафика, особенно по мультиаендорным сетям. Он также позволяет вещателям внедрять унифицированные сторонние инструменты мониторинга по всей своей инфраструктуре, позволяя измерять общим методом больший набор параметров.

Чтобы побудить производителей принять новый стандарт, необходимо продемонстрировать потенциальные преимущества их продуктов в случае поддержки нового стандарта. Поэтому группа EBU-IPM должна обеспечить вещателям программный инструмент, который может контролировать медиа трафик с использованием нового стандарта.

Программное обеспечение должно быть независимым от любой операционной системы, производителя оборудования и патентованных технологий. Иначе есть риск, что новый стандарт будет восприниматься связанным с определенным поставщиком.

Отсутствие подходящих безлицензионных прс обмен сетевой информацией :N-IPM заказать разработку собственного программного обеспечения, EBU IPM SNMP Stream Monitor (EisStream). Задача состояла в разработке полностью автоматизированного инструмента сетевого мониторинга с открытым источником, на основе стандарта и независимого от платформы.

Как видно на Рис. 2, EisStream предназначена для мониторинга трафика данных на множестве уровней модели OSI. Программа выполняет все функции обнаружения, анализа и мониторинга путем обработки небольшого набора общих данных MIB, извлеченных из различных сетевых устройств через SNMP (Simple Network Management Protocol).

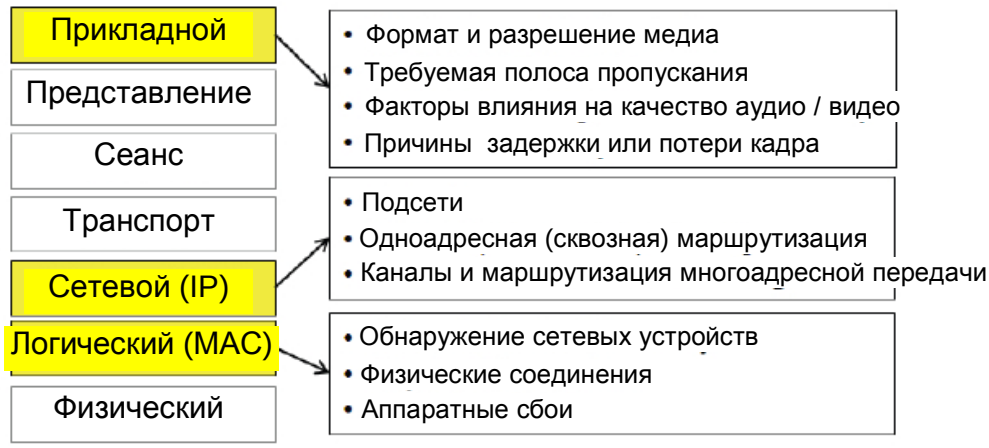

Рис. 2: Уровень мониторинга EisStream

[2] EBU Integrated Monitoring Solution for Media Streams on IP Networks, http://eisstream.sourceforge.net/

Вследствие того, что разработка EisStream шла параллельно с проектом EBU Tech 3345, первая часть выпуска программного обеспечения состоит только из функций обнаружения и анализа сети на Layer 2 и 3 с использованием существующих стандартных объектов MIB, общепринятых всеми крупными производителями. В Таблице 1 приведен список MIB объектов, используемых в программе EisStream.

Таблица 1: Поддержка MIB, требуемая EisStream

MIB	Таблица	Объект	Требуемая поддержка
MIB-II	(Leaf Objects)	sysDescr	Все устройства
		sysServices	
		ipForwarding	
	ifTable	ifIndex	
		ifDescr	
		ifPhysAddress	
	ipAddrTable	ipAdEntAddr	
		ipAdEntIfIndex	
		ipAdEntNetMask	
	ipRouteTable	ipRouteDest	
		ipRouteIfIndex	
		ipRouteNextHop	
		ipRouteType	
	ipNetToMediaTable	ipNetToMediaIfIndex	
		ipNetToMediaPhysAddress	
		ipNetToMediaNetAddress	
IP-FORWARD-MIB	ipCidrRouteTable	ipCidrRouteDest	Только маршрутизаторы
		ipCidrRouteMask	
		ipCidrRouteNextHop	
		ipCidrRouteIfIndex	
		ipCidrRouteType	
BRIDGE-MIB	dot1dTpFdbTable	dot1dTpFdbAddress	Многоуровневые коммутаторы и мосты
		dot1dTpFdbPort	
		dot1dTpFdbStatus	
IPMROUTE-STD-MIB	ipMRouteTable	ipMRouteSource	Многоадресные маршрутизаторы
		ipMRouteSourceMask	
		ipMRouteUpstreamNeighbor	
		ipMRouteInIfIndex	
		ipMRouteAddress	
		ipMRouteRtMask	
	ipMRouteNextHopTable	ipMRouteNextHopIfIndex	
		ipMRouteNextHopState	
	ipMRouteScopeNameTable	ipMRouteScopeNameString	

EisStream предназначена для обеспечения универсальной платформы с открытым источником, основанной исключительно на SNMP, чтобы вещатели и производители реализовали новые объекты MIB для мониторинга, предложенные в EBU Tech 3345. Поэтому после полного принятия EBU Tech 3345 программа может легко обновляться для обеспечения информации сетевого мониторинга на прикладном уровне, обеспечивая единое решение для мониторинга мультивендорной сети.

Исходные коды программы **EisStream** выпущены в SourceForge: (http://eisstream.sourceforge.net/).

Текущая версия EisStream обеспечивает пять основных функций:

- Device Discovery – обнаружение всех устройств в неизвестной сети
- Physical Topology – идентификация всех физических соединений в этой сети
- IP Layer Information – сбор маршрутной информации подсети и Layer 3
- End-to-End Physical Path – отслеживание всех узлов, включенных в передачу IP пакета
- Multicast Maps – обнаружение и преобразование структур маршрутизации для всех каналов

2.1 Device Discovery

EisStream может начинать обнаружение сети из любого SNMP-совместимого хоста и обнаруживать все устройства в неизвестной сети, при условии, что вся сетевая инфраструктура поддерживает SNMP и их строки сообщества SNMP известны.

Эта функция не только позволяет полностью автоматизированный анализ совершенно неизвестной сети, но и чрезвычайно полезна в управлении сетей, топологии которых уже известны, т.к. в крупномасштабных сетях все труднее поддерживать обновление информации вручную.

Поскольку сообщение SNMP передается только в виде UDP пакета на IP уровне, оно может отправляться только на известный IP адрес. Поэтому программа начнет процесс обнаружения с передачи первых сообщений SNMP либо на локальный адрес 127.0.0.1, либо на указанный пользователем IP адрес. После получения информации о первом устройстве EisStream будет исследовать IP адреса его логических соседей. Программа повторяет этот процесс, пока не обнаружит все устройства в сети.

Как показано на Рис. 3, если пользователю не нужно менять начальный IP адрес или не используется пароль SNMP не по умолчанию, EisStream не потребует первичного ввода для обнаружения неизвестной сети.

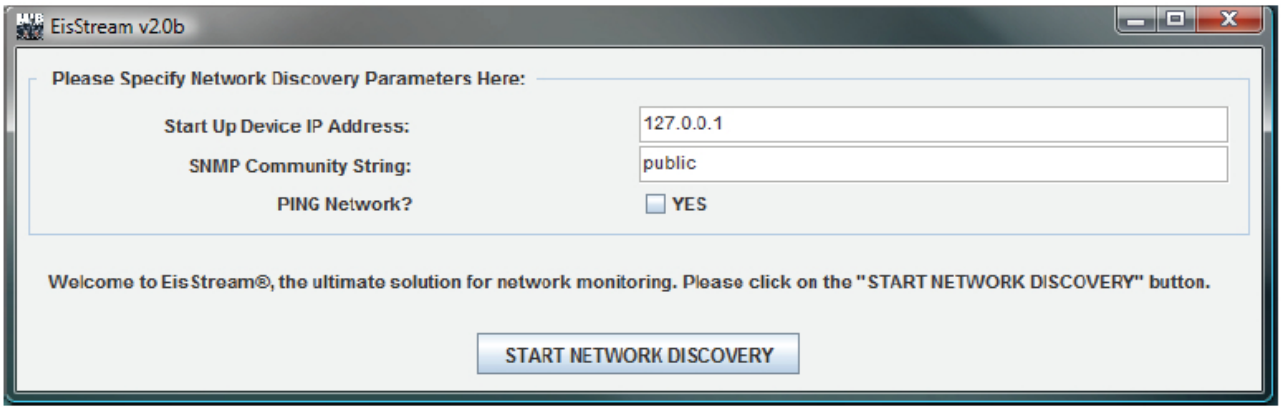

Рис. 3: Экран запуска EisStream

Обнаруженные устройства, включая устройство запуска, классифицируются по трем основным группам.

- Маршрутизаторы: устройства, передающие далее IP пакеты. Маршрутизаторы можно идентифицировать, считав значение объекта “ipForwarding” в MIB-II. Следовательно, Gateway PC также считаются маршрутизаторами. Маршрутизаторы с мостовыми (немаршрутизированными) портами известны как многоуровневые коммутаторы.
- Мосты: часто называются коммутаторами Layer 2. Это устройства, которые не передают далее IP пакеты и не имеют ввода данных в “ipRouteTable”.
- Хосты: подключенное к сети аудиовизуальное оборудование и устройства, включая PC и рабочие станции, которые не передают далее IP пакеты, но имеют ввод данных в “ipRouteTable”.

Подробный процесс обнаружения устройств см. в § 4, Алгоритмы.

2.2 Physical Topology

EisStream может автоматически генерировать топологию физической сети, анализируя информацию, собранную на этапе обнаружения сети. По той же причине, что и Device Inventory, эта функция сэкономит администраторам сетей массу времени и усилий либо на ознакомление с новой сетью, либо на повседневное управление имеющимися сетями. Самое главное, что эта функция также позволяет легче обнаруживать любые изменения физических соединений, обеспечивая эффективный и точный анализ первопричин сбоев в сети.

Вообще, каждый хост в сети соединен с маршрутизатором или с мостом, который соединен с другими маршрутизаторами или мостами. Поэтому хосты считаются «листьями» в физической топологии, а маршрутизаторы и мосты – «ветвями».

EisStream анализирует физическую структуру сети, определяя сначала соединения между ветвями. После определения топологии ветвей к ним можно просто прикрепить листья.

EisStream идентифицирует тип соединения между двумя любыми физическими соседями как одно из:

- “L2-L2”, между двумя мостовыми портами в мостах и многоуровневых коммутаторах.
- “L2-L3”, между мостовым и маршрутизированным портами.
- “L3-L3”, между двумя маршрутизированным портами.

Тип соединения любого данного порта идентифицируется проверкой числа записей Address Resolution Protocol (ARP) этого порта и соответствующих полей в Forwarding Information Base (FIB) устройства. Тип соединения порта должен повторяться другим портом, который с ним соединен.

Детали методов анализа типов соединений, определяющего физическое соседство и устанавливающие топологию сети, описаны в п.4 Алгоритмы.

2.3 IP Layer Information

EisStream также может обеспечивать точную информацию об IP подсетях и маршрутизации Layer 3. Сейчас она полностью поддерживает все спецификации IPv4, включая бесклассовую адресацию и нуль подсети. Эти функции также могут быть расширены для поддержки IPv6, хотя на момент написания (апрель 2011) не проводилось никаких тестов для подтверждения.

Наличие набора исчерпывающей информации Layer 3 необходимо для мониторинга IP сетей. Администраторы сетей опираются на эту информацию в нахождении отдельных устройств и выполнении задач высшего уровня, например, мониторинга и конфигурирования.

Как говорилось выше, помимо обнаружения устройств, EisStream собирает полный набор сетевых данных, связанных с устройствами. Информацию Layer 3 о каждом сетевом порте устройства можно найти в этих данных. IP подсети идентифицируются вычислением сетевых адресов из IP адресов и соответствующих значений маски подсети. Маршрутную информацию Layer 3 для маршрутизаторов можно найти в данных их виртуальных интерфейсов.

Это относительно развитая функция инструментах сетевого мониторинга; поэтому не требуется дальнейших деталей для описания методов, используемых EisStream в сборе информации IP уровня.

2.4 End-to-End Physical Path

Одна из важнейших характеристик EisStream – способность отслеживать сквозной физический тракт между любыми двумя хостами в сети и идентифицировать номер каждого отдельного порта всех устройств, входящих в цепь соединения.

Мониторинг медиа потока между двумя хостами сейчас ограничен только прикладным уровнем. Все физические узлы, такие как мосты и маршрутизаторы, участвующие в установке соединения между хостами, прозрачны для большинства инструментов мониторинга в результате общих характеристик многоуровневого принципа связи. Многие специальные аппаратные или интегрированные программные инструменты, предоставляемые производителями сетевого оборудования, могут контролировать все уровни, но они либо патентованные (и потому очень дорогие), либо применимы только к устройствам того же производителя.

Как показано на Рис. 4, традиционные инструменты мониторинга могут давать только информацию о потоке данных на прикладном уровне, представленном на рисунке синей линией. С новым стандартом EBU MIB и EisStream теперь возможно последовательное отслеживание на уровнях IP и MAC, как показано зеленой линией, представляющей физический тракт пакета данных. Эта функция позволяет EisStream идентифицировать и контролировать все узлы тракта.

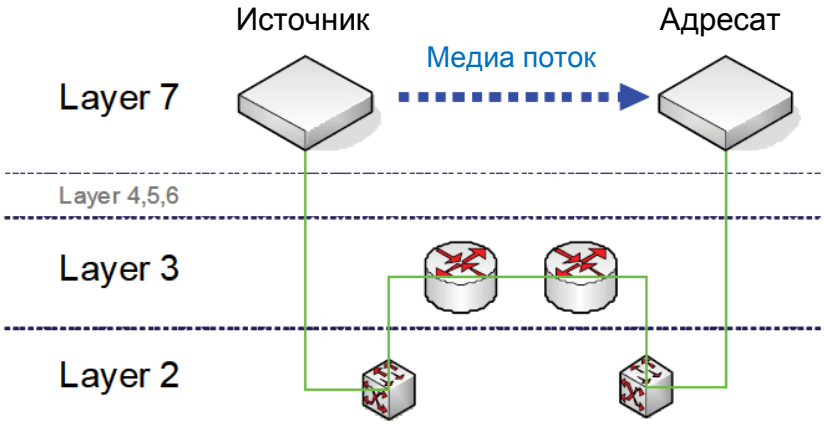

Рис. 4: Физический тракт медиа потока

Используя информацию IP уровня, собранную в процессе обнаружения, EisStream сначала определяет логические маршруты между двумя хостами. Если они находятся в разных подсетях, EisStream идентифицирует все маршрутизаторы, установившие соединение Layer 3 между подсетями. Наконец, физический тракт между пограничными маршрутизаторами и хостами преобразуется из физической топологии.

Подробно функция отслеживания сквозного физического тракта EisStream объясняется в § 4, Алгоритмы.

2.5 Multicast Maps

Еще одна уникальная характеристика EisStream – способность обнаружения многоадресных потоков и их маршрутных структур. Для каждого обнаруженного канала / потока многоадресной передачи EisStream также преобразует все маршрутизаторы и их место в передающей структуре данного канала / потока. Используя эту функцию вместе с функцией отслеживания сквозного физического тракта, EisStream может контролировать трафик и физический тракт между любым многоадресным источником и клиентом.

Технология многоадресной передачи широко используется для автоматической маршрутизации и протоколов сетевого управления, таких как Routing Information Protocol (RIP), Open Shortest Path First (OSPF) и Network Time Protocol. Она также широко применяется вещателями для передачи видео и аудио по IP в цепях контрибуции и распространения.

EisStream собирает всю информацию многоадресной передачи с маршрутизаторов во всей сети на этапе обнаружения. Сначала она идентифицирует все многоадресные маршрутизаторы, а затем по содержащейся в них информации определяет каналы многоадресной передачи и их адреса. Каждый канал содержит ряд хостов как источников и клиентов и маршрутизаторы, образующие ветви определенного дерева многоадресной передачи.

Маршрутизатор в канале многоадресной передачи может войти в состояние, когда он не передает никакого трафика, т.к. ни один клиент не подписан на этот канал, как показано на Рис. 5. EisStream все равно включит этот маршрутизатор в многоадресное дерево в качестве ветви, но без дальнейших ветвей и листьев, в то время как другие инструменты сетевого мониторинга, основанные на сдвиге трафика, исключат такие маршрутизаторы из канала.

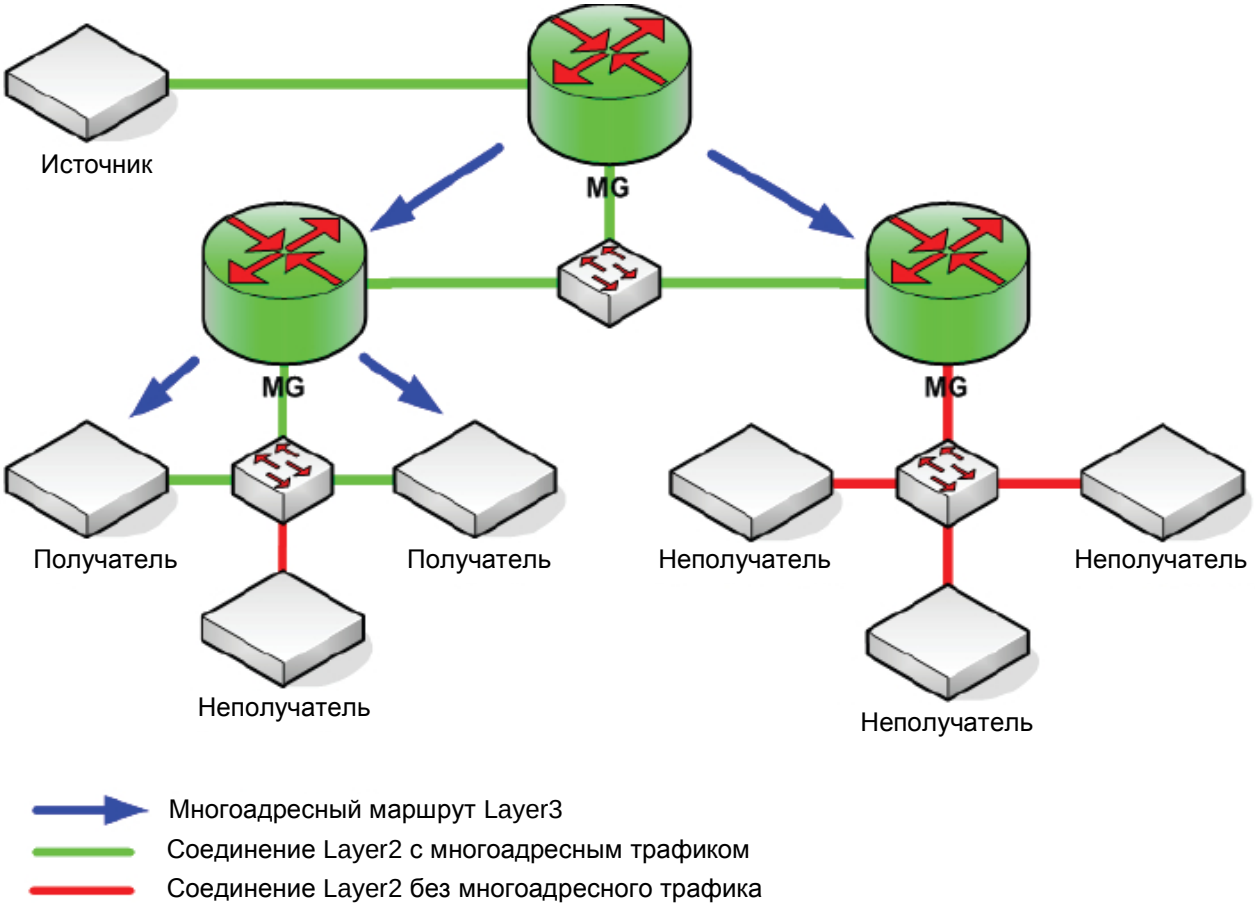

Рис. 5: Бездействующий маршрутизатор в дереве многоадресной передачи

EisStream может обнаруживать бездействующие маршрутизаторы в канале многоадресной передачи, т.к. собирает информацию независимо от ручных конфигураций, например, режима PIM, версии IGMP, приоритета инициализации маршрутизаторов и установок Rendezvous-Point. Другими словами, программа узнает только реальное поведение маршрутизаторов. Поэтому, пока маршрутизатор передает информацию канала многоадресной передачи, он считается частью маршрутной структуры, независимо от того, активен он или нет.

Дальнейшие подробности того, как EisStream обнаруживает и анализирует каналы многоадресной передачи, можно найти в § 4, Алгоритмы.

3. Архитектура программного обеспечения

Программа EisStream разработана на Java для независимости от платформы и потенциальной реализации в качестве веб-услуги. Этот раздел даст разработчикам Java общее введение в структуру программы, включая классы общих пакетов и ключевые методы каждого класса в исходном коде EisStream.

В целях выпуска с открытым источником программа включает очень ограниченное число безлицензионных внешних пакетов:

- SNMP4J: используется для отправки и получения сообщений SNMP.
- JGraphT: используется для генерирования графических интерактивных сетевых диаграмм.
- JGraph: требуется для JGraphT.

Помимо вышеперечисленных пакетов, EisStream имеет три внутренних пакета, разработанных полностью внутри BBC.

- **NMS**: (Network Monitoring Station) управляет связью с устройствами в сети.
- **NETWORK**: отвечает за все функции обнаружения и анализа в сети.
- **GUI**: генерирует и управляет функциями пользовательских интерфейсов.

Базовые функции программы EisStream полностью содержатся в пакете NETWORK, а NMS является интерфейсами с сетью и пользователями. Внешние пакеты используются просто для управления базовым графическим отображением и сетевыми сообщениями. На Рис. 6 показаны отношения между всеми пакетами EisStream.

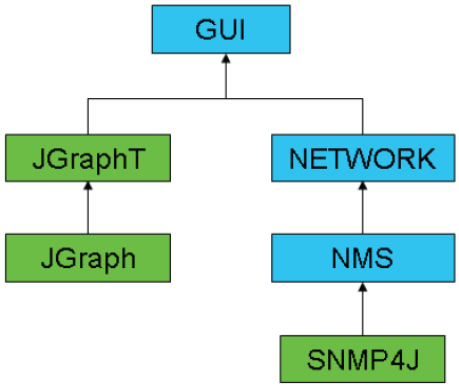

Рис. 6: Структура зависимости пакетов EisStream

3.1 Пакет NMS

В типичном сценарии мониторинга SNMP Network Monitoring Station (NMS) связывается с устройствами сети через сообщения SNMP. Обычно NMS является физическим сервером, который передает и принимает пакеты UDP, содержащие строки сообщений в форме десятичных представлений. Кроме того, пакет NMS в EisStream также отвечает за скремблирование строки сообщения SNMP из файла MIB, подтверждение и извлечение данных из ответных строк и передачу других типов сообщений, например, пинговых.

В пакете NMS EisStream пять объектов.

- NMS.java: публичный класс для скремблирования запросов SNMP и интерпретации ответов SNMP.
- MibFileAccessor.java: приватный класс для нахождения строки сообщения SNMP из стандартного файла MIB.
- SnmpMessage.java: приватный класс для содержания одного объекта сообщения SNMP для использования SNMP4J.
- SnmpTransponder.java: приватный класс для побуждения SNMP4J к отправке и приему сообщений SNMP.
- IcmpSender.java: приватный класс для передачи одного пакета Ping на данный IP адрес.

На Рис. 7 показаны отношения между объектами в пакете NMS. Объект NMS эксклюзивно использует все приватные классы в пакете для обмена сообщениями SNMP с сетевыми устройствами. Это класс, используемый другими пакетами для извлечения данных MIB из сетевых устройств. Использование объекта IcmpSender объясняется в § 4, Алгоритмы.

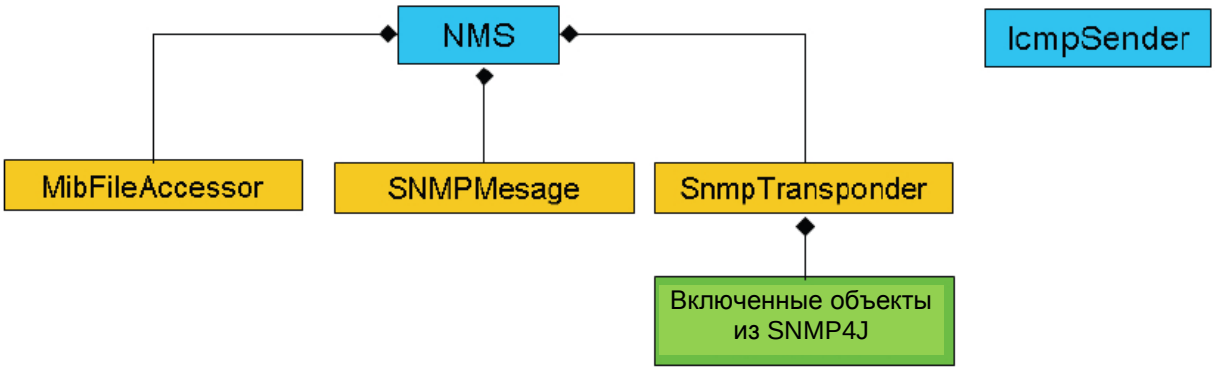

Рис. 7: Упрощенная структура объектов UML пакета NMS

После команды на отправку сообщения SNMP на IP адрес объект NMS сначала вызывает “MibFileAccessor.getSnmpMessageByName(String)” для получения содержимого сообщения SNMP из локальной библиотеки файлов MIB и сохраняет его как объект SnmpMessage.

Затем объект NMS передает объект сообщения объекту SnmpTransponder и вызывает “SnmpTransponder.SendSnmpMessage(SnmpMessage)” для передачи и приема сообщения SNMP с IP адреса. Получив ответное сообщение, объект NMS подтверждает данные и использует метод “validSnmpReply()” для индикации приемки данных.

3.2 Пакет NETWORK

Пакет NETWORK – центр вычислений и данных программы EisStream, отвечающий за все ее функции. Он не имеет прямого интерфейса с пользователями и сетями, что в будущем облегчает его реализацию в виде веб-услуги.

В пакете EisStream NETWORK 10 объектов. Все они – публичные классы, т.к. должны полностью взаимодействовать с другими частями программы.

- Interface.java: представляет физический порт устройства;
- Device.java: представляет общий объект, который может быть хостом, маршрутизатором или мостом;
- Switch.java: наследуя Device.java, представляет общий объект, который может быть мостом или многоуровневым коммутатором, оба из которых – самые распространенные типы современных маршрутизаторов;
- Router.java: наследуя Switch.java, представляет маршрутизатор, который также может быть многоуровневым коммутатором;
- Discoverer.java: выполняет функцию обнаружения сетевых устройств;
- Layer2Map.java: выполняет функцию анализа физической топологии;
- Layer3Map.java: выполняет функцию сбора информации Layer 3;
- Path.java: выполняет функцию отслеживания сквозного физического тракта;
- MulticastGroup.java: представляет структуру многоадресной передачи, состоящую из многоадресных маршрутизаторов, источников и физических портов этих устройств;
- MulticastMap.java: выполняет функцию преобразования каналов многоадресной передачи;

На Рис. 8 показаны отношения между объектами в пакете NETWORK. Классы Interface и Device – два принципиальных компоновочных блока всех функций, т.к. обычно все сети состоят из устройств, соединенных через порты. Три основных типа устройств представлены объектами Device, Switch и Router. Каждая базовая функция EisStream также определяется как отдельный объект в пакете NETWORK. MulticastGroup – это объект данных, необходимый для представления особых структурных отношений группы устройств.

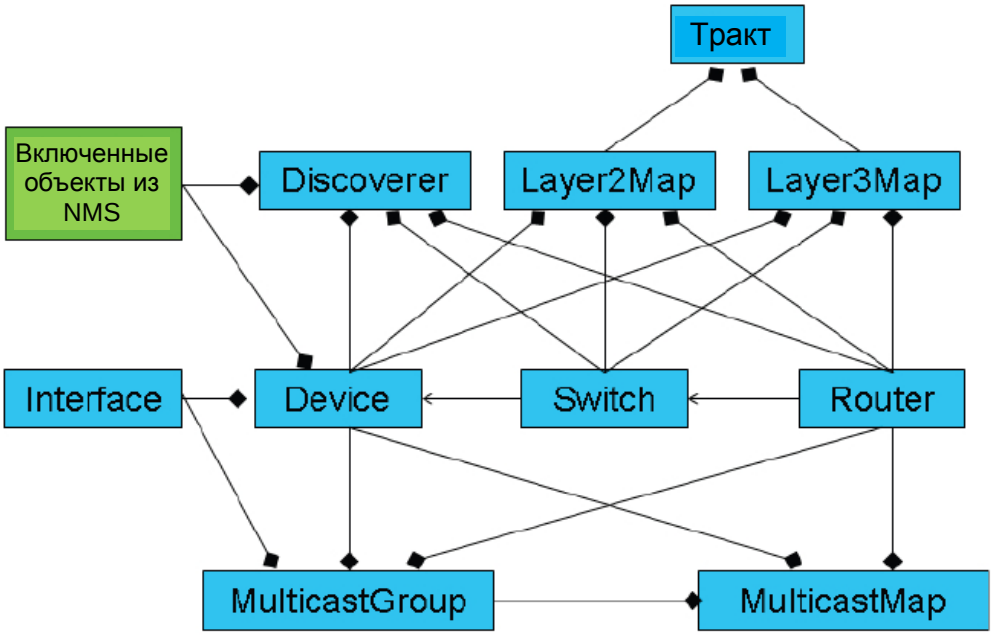

Рис. 8: Упрощенная структура объектов UML пакета NETWORK

Метод выдачи команды пакету NMS на отправку и получение сообщений SNMP от устройств находится в объекте Device и также наследуется объектами Switch и Router. Это позволяет связывать передачу сообщений SNMP из определенных MIB с соответствующим типом устройства, например, только объект Switch может передавать сообщения SNMP из BRIDGE-MIB. Поэтому в будущем, когда BRIDGE-MIB изменится, нужно будет обновить только объект Switch.

Объект Discoverer внедряет одиночный процесс через метод “DiscoverAll(boolean)”. Он создает новый объект Device каждый раз, когда новый IP адрес узнается рекурсивными методами “nextHopDiscovery(Device)” или “arpCacheDiscovery(Device)”. Затем новый объект Device заполняется методом “Device.populate()”. Объект Interface создается и заполняется внутри объекта Device для каждого физического порта (который он обнаружил??). Затем Discoverer будет использовать методы “isLayer2Switch(Device)” и “isLayer3Router(Device)” для идентификации Device как Host, Switch или Router, и помещения его в соответствующий класс по необходимости. Объекты Switch и Router имеют дальнейшие поля для заполнения, определяемые функциями маршрутизации и мостовых соединений. Алгоритм цикличного обнаружения продолжается до тех пор, пока не обнаружится новый IP адрес. Затем программа выходит из процесса обнаружения.

После вызова метода “update(ArrayList<Router>, ArrayList<Switch>, ArrayList<Device>)” объект Layer2Map принимает обнаруженный набор маршрутизаторов, коммутаторов и хостов и находит ядро сети методом “findCore(ArrayList<Router>)”. Затем он запускает различные алгоритмы для анализа физического соединения методом “buildMap()”, который вызывает рекурсивные методы “expand(Switch)” и “findBranchConnections(Interface)” для установления всех соединений-ветвей среди маршрутизаторов и коммутаторов. Затем устанавливаются соединения-листья для хостов методом “findLeafConnection(Interface)”. По окончании анализа объект Layer2Map подтверждает результаты, вызывая метод “validateConnections()”. Есть также метод “getPath(Interface, Device, Interface, ArrayList<Interface>)”, который применяется для нахождения физического тракта между двумя устройствами.

Layer3Map также имеет метод “update(ArrayList<Router>, ArrayList<Device>)”, который принимает данные из объекта Discoverer для обработки информации Layer 3. Кроме метода “buildMap()”, который производит HashMap подсетей и их членов, Layer3Map также обеспечивает метод “routeSubnets(String, String)” для нахождения IP маршрута среди различных подсетей, что полезно для отслеживания сквозного физического тракта. Объект Layer3Map также обеспечивает набор статичных функций для обработки IP информации, например, “findIPMask(String)”, “findNetworkNumber(String, String)” и “sameSubnet(Interface, Interface)”.

Объект Path использует объекты Layer2Map и Layer3Maps и находит физический тракт между двумя IP адресами, вызывая метод “getPath(String, String)”. Используя сначала “Layer3Map.routeSubnets(String, String)” для обнаружения узлов Layer 3, а затем “Layer2Map.getPath(Interface, Device, Interface, ArrayList<Interface>)”, объект Path находит весь физический тракт между каждой парой узлов Layer 3. Объединяя эти тракты согласно порядку узлов Layer 3, объект Path возвращает полный физический тракт в виде индексированного HashMap интерфейсов и устройств.

Как и объекты Layer2Map и Layer3Map, объект MulticastMap использует “update(ArrayList<Router>, ArrayList<Device>)” и “buildMap()” для приема и обработки данных для обнаружения групп многоадресной передачи и их структур. Он хранит всю информацию группы многоадресной передачи в объекте multicastGroup и сохраняет список этих объектов для всей сети.

3.2 Пакет GUI

Пакет GUI разработан для управления графическим представлением всей сетевой информации на разных уровнях. Главный пользовательский интерфейс имеет окно, содержащее три вкладки для отображения физической топологии, подсетей и многоадресной передачи. Есть также всплывающее окно для отображения детальной информации и пользовательских сообщений.

В пакете GUI 6 объектов.

- Main.java: запускает и выключает программу EisStream.
- GUIWindow.java: создает объект главного пользовательского окна.
- L2MapTab.java: создает вкладку для интерактивной карты физической топологии
- L3MapTab.java: создает вкладку для интерактивной таблицы информации IP Subnet
- MulticastMapTab.java: создает вкладку для карт каналов многоадресной передачи.
- InfoWindow.java: создает объект всплывающего окна.

На Рис. 9 показаны отношения между объектами в пакете GUI.

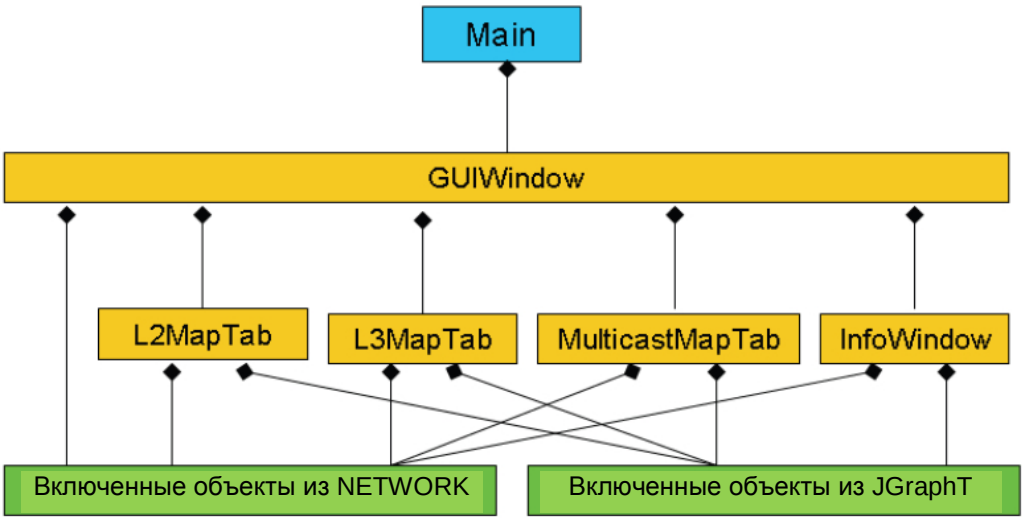

Рис. 9: Упрощенная структура объектов UML пакета GUI

Класс Main прежде всего отвечает за внедрение GUIWindow, которое должно начинаться из другого класса. Как только начато окно, объект GUIWindow берет на себя роль драйвера программы EisStream. До реализации функций прямого мониторинга вкладки и всплывающее окно присутствуют только для представления информации пользователю. Функции обнаружения и анализа запускаются исключительно объектом GUIWindow.

При запуске объект Main создает внутри себя объект GUIWindow. Затем объект GUIWindow берет несколько значений параметров от пользователя, который затем начинает процесс обнаружения и анализа. По окончании объект GUIWindow создаст объекты L2MapTab, L3MapTab и MulticastMapTab в окне пользователя. Во вкладках есть цветные квадратики в JGraphT для представления разных типов устройств. После нажатия на квадратик устройства создается объект InfoWindow, после того как объект вкладки вызовет метод “displayInfo()”.

4. Алгоритмы

Вследствие требования, что EisStream должна иметь открытый источник и что нельзя использовать патентованной библиотеки, и из-за отсутствия подходящих безлицензионных пакетов все базовые функции EisStream были разработаны с нуля. В результате было разработано много оригинальных новаторских методов и алгоритмов.

Кроме функции для информации Layer 3, все остальные характеристики EisStream обеспечиваются оригинальными методами и алгоритмами.

4.1 Device Discovery

Чтобы начать обнаружение сети, EisStream должна получить IP адрес любого устройства, поддерживающего SNMP. Если этот адрес не указан, программа также может начать с адреса закольцовывания локальной машины, на которой она работает, при условии, что эта машина поддерживает SNMP.

На Рис. 10 показан весь процесс обнаружения.

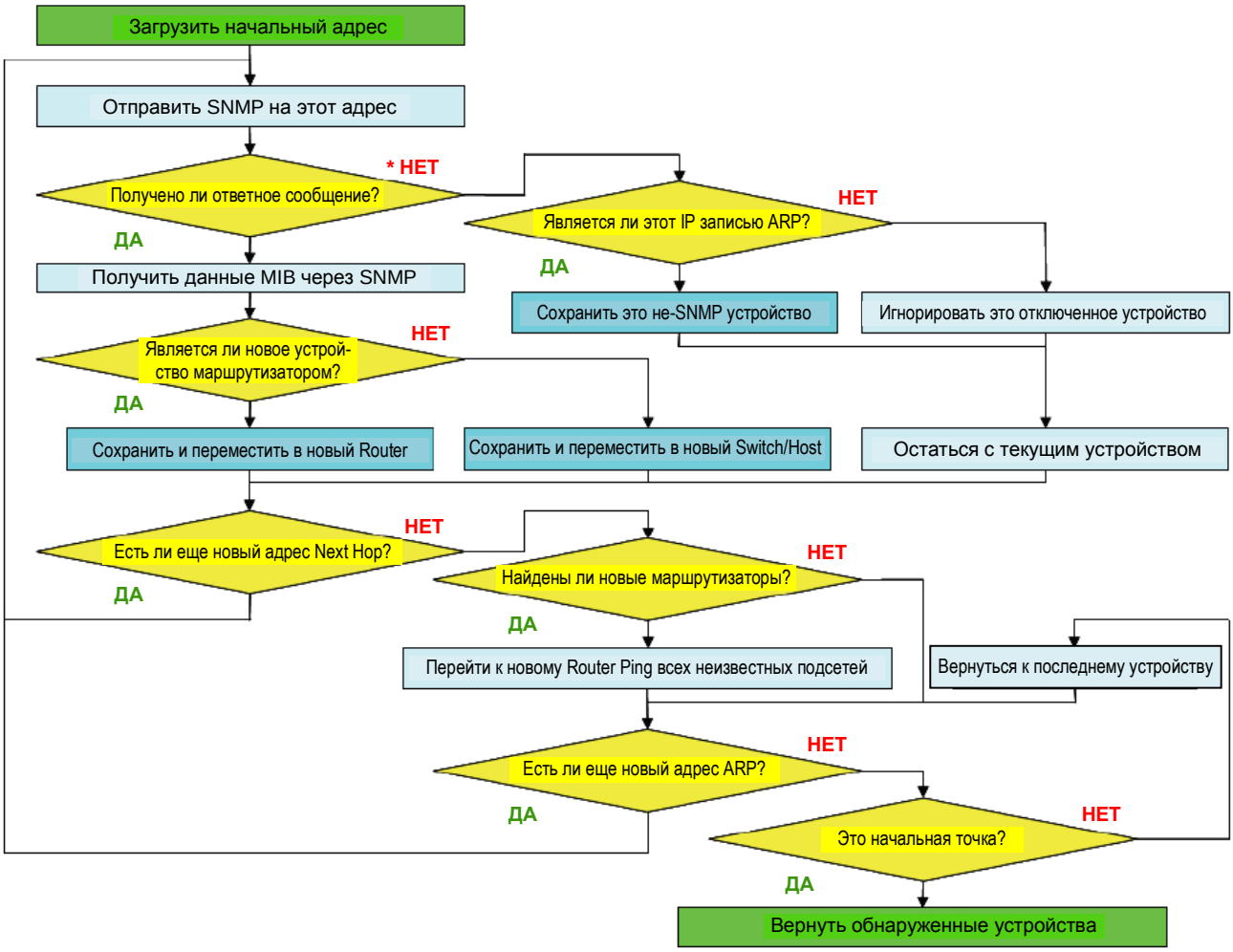

Рис. 10: Алгоритм обнаружения

Для обнаружения новых устройств используются две группы объектов MIB: “ipRouteTable” и “ipNetToMediaTable”. “ipRouteTable” содержит маршрутную информацию Layer 3, также известную как Next Hop, которая используется для обнаружения IP адресов шлюза-маршрутизатора устройства. После обнаружения нового маршрутизатора его Next Hop ведет к обнаружению другого маршрутизатора. “ipNetToMediaTable” содержит записи ARP, используемые для обнаружения IP адресов других логических соседей устройства. Используя запись ARP всех устройств, рекурсивный алгоритм потенциально может работать по всей сети.

Обнаружение через Next Hop предпочтительнее ARP, т.к. ведет к обнаружению новых маршрутизаторов и потому имеет больше шансов обнаружить больше подсетей, что ведет к обнаружению большего числа устройств.

Когда обнаружено новое устройство, либо через Next Hop, либо через ARP с предыдущего устройства, первым делом надо всегда смотреть на данные Next Hop текущего устройства. Обнаружение через ARP начинается только тогда, когда не найдено ни одного нового маршрутизатора через Next Hop.

Когда обнаруженный IP адрес не отвечает на SNMP, возможно, что устройство либо не поддерживает SNMP, либо отсоединено. Если этот IP адрес – Next Hop, показывающий, что устройство является маршрутизатором, то вероятнее, что маршрутизатор отсоединен, т.к. для него очень странно не поддерживать SNMP. По той же причине, что будет указана в следующей главе, если устройство обнаружено через ARP, то вероятнее, что это подключенное устройство без поддержки SNMP.

EisStream имеет определенную степень допуска для устройств, не поддерживающих SNMP. Вместо возврата ошибки она создает «фиктивное» устройство с «фиктивным» интерфейсом для устройства не-SNMP. Это устройство считается хостом, т.к. правильный тип невозможно идентифицировать без получения информации об устройстве через SNMP.

В типичной вещательной сети устройства скорее всего часто обмениваются друг с другом обновленной информацией о своих логических соседях. Этот сценарий идеален для использования EisStream, которая зависит исключительно от сбора сетевой информации от отдельных устройств через сообщения SNMP.

В определенных случаях, когда сетевое устройство не действует в течение долгого периода времени, информация о его логических соседях часто удаляется в результате функции уборки в устройстве. Это представляет проблему для обнаружения всей сети программой EisStream. Если программа не сможет искусственно сгенерировать некоторый трафик в или из IP адресов этих устройств, эти устройства никогда не будут обнаружены программой.

Эта проблема также имеет прямое влияние на обнаруживаемость мостов, которые в нормальных операциях прозрачны для уровня IP. Если IP адрес прозрачного моста не виден в списке логических соседей какого-либо устройства, программа не сможет их обнаружить.

Для решения этой проблемы на каждый неизвестный адрес известной подсети последовательно передается один пинг-пакет (Internet Control Message Protocol, ICMP). Это обновляет запись ARP на короткий период времени, не перегружая сеть.

Учитывая, что большинство вещательных сетей имеют отдельные подсети управления с должным отделением от производственных подсетей, рекомендуется запускать EisStream в режиме “Ping-Enabled” в подсети управления. Можно ограничить адресаты ICMP только сетью управления. Это дает абсолютную гарантию, что производственной сети не помешает никакой лишний трафик.

4.2 Physical Topology

Как уже говорилось, все хосты считаются листьями, а все мосты и маршрутизаторы – ветвями любой IP сети. Следовательно, каждый мост или маршрутизатор считается «точкой разветвления».

Для установления полной топологии сети нужно сначала определить соединения между всеми точками разветвления.

В общем масштабе нужен алгоритм для рекурсивного измерения от одной точки разветвления до другой, пока не будет охвачена вся сеть. В микроскопичном масштабе нужно несколько других методов для определения типа физического соединения порта и нахождения, куда подключен другой порт.

4.2.1 Общий алгоритм для разветвленной топологии

Процесс нахождения разветвленной топологии начинается с определения гипотетического «ядра» сети, который является маршрутизатором – самым популярным адресатом HextNext Hop. Чем больше к устройству направляется других маршрутизаторов, тем вероятнее, что она находится в центре сети.

На Рис. 11 показан алгоритм, написанный на языке Java для нахождения ядра сети. Это не обязательно центральный маршрутизатор сети, но служит хорошей отправной точкой.

```
int max_count = 0;
router core = new router ();

for (every CURRENT_Router) {
      int current_count = 0;
      for (every OTHER_Router) {
          for (every "ipRouteNextHop" entry) {
              if ("ipRouteType" = indirect) {
                  if (CURRENT_Router's IP address list.contains("ifRouteNextHop")) {
                      ++current_count;
                  }
              }
          }
      }
      if (current_count > max_count) {
          core = CURRENT_Router;
          max_count = current_count;
      }
}
return core;
```

Рис. 11: Алгоритм для нахождения гипотетического ядра сети в синтаксисе Java.

С выбранным ядром вокруг него может быть выстроена структура соседства путем нахождения непосредственно связанных точек разветвления. Для каждого из соседей будет продолжен тот же процесс, пока не будут найдены все точки разветвления в соседстве с этим ядром Есть также функция “catch-all” для точек разветвления, недоступная из ядра. Каждая из этих точек разветвления расширяется для охвата других точек, уже найденных в соседстве с ядром, как показано на Рис. 12.

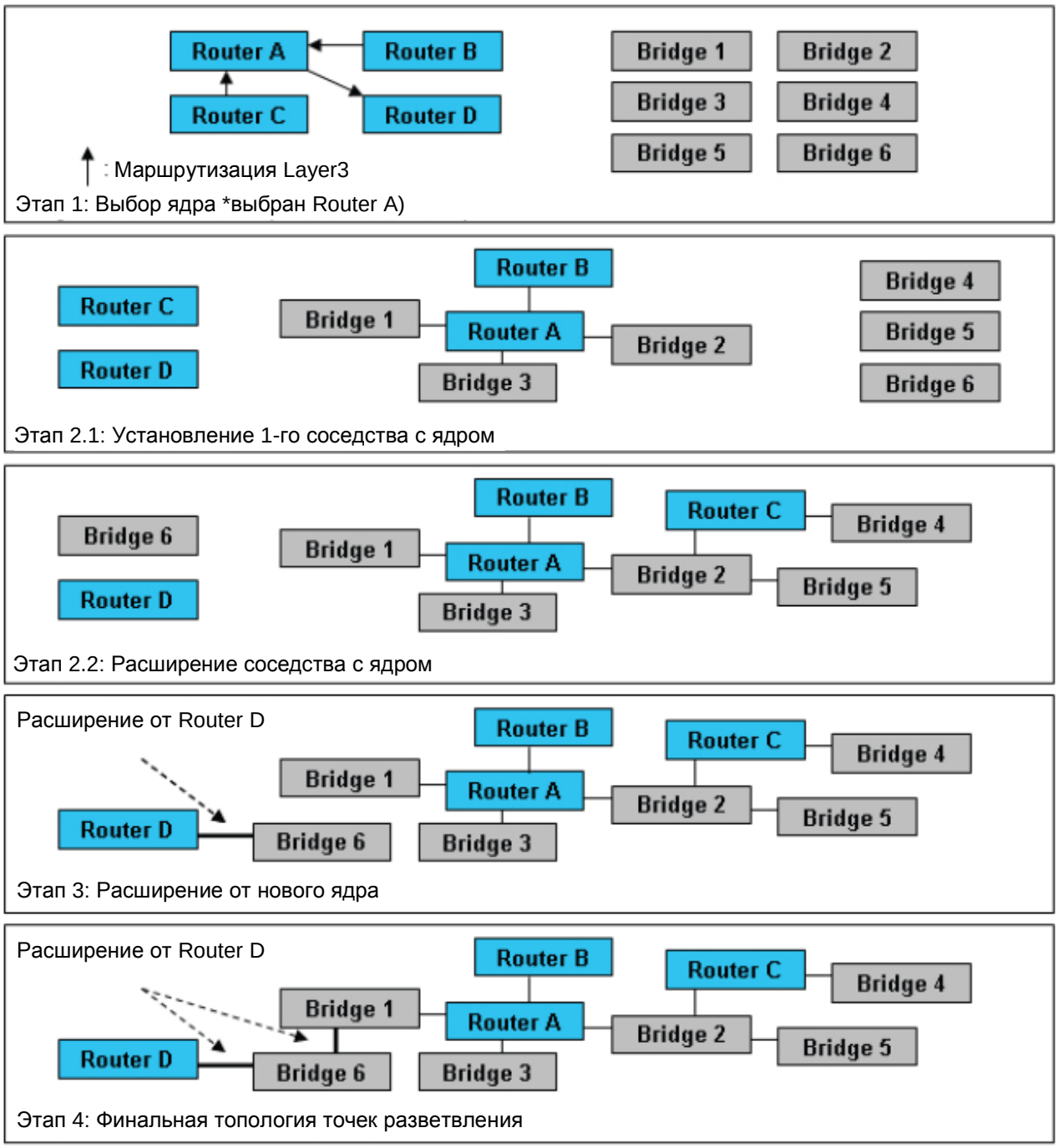

Рис. 12: Пример нахождения топологии точек разветвления

4.2.2 Специальные методы для соединений точек разветвления

Магистраль сети состоит из межсоединенных точек разветвления. Как уже говорилось, есть три типа физических соединений; следовательно, любые два физически соединенные порта должны иметь соответствующие типы соединения.

Для нахождения подключенного порта для любого данного порта выбираются все остальные порты устройства с соответствующими типами соединения. Затем можно провести среди кандидатов анализ соединения, подходящий для данного типа, чтобы найти фактически подключенный порт.

4.2.2.1 Определение типа соединения

Алгоритм для проверки физического соединения начинается с анализа вероятного типа соединения порта. На основе типа соединения применяется соответствующий алгоритм для нахождения физического соседа данного порта.

Используя логику, показанную в Таблице 2, можно точно определить тип соединения порта в маршрутизаторе или мосте.

Таблица 2: Таблица решений для определения типа соединения

Тип соединения		**Кол-во полей в FIB**		
		0	**1**	**n**
Кол-во полей ARP	0	n/a	L2-L2	L2-L2
	1	L3-L3	L2-L3	L2-L2
	n	L3-L2	L2-L2	L2-L2

Например, если порт не присутствует в Forwarding Information Base (FIB) MAC адресов, то вероятно, что он маршрутизирован. Затем посмотрим на запись ARP этого порта, Если там много полей данных, то порт, вероятно, соединен с портом коммутатора другой точки разветвления. Следовательно, соединение с этим портом, вероятно, является соединением L3-L2.

4.2.2.2 Анализ соединений L2-L2

Как показано на Рис. 13, FIB моста или многоуровневого коммутатора определяет порт, в который должен передаваться пакет Layer 2 с внешним MAC адресом адресата. Данные FIB содержатся в “dot1dTpFdbTable” в BRIDGE-MIB, с внешним MAC адресом адресата пакета Layer 2, хранящегося в объекте “dot1dTpFdbAddress”, и индексом направляемого порта в “dot1dTpFdbPort”.

Набор всех полей “dot1dTpFdbAddress” моста с одинаковым “dot1dTpFdbPort” – это коллекция всех MAC адресов адресата всех пакетов Layer 2, передаваемых этим портом.

С другой стороны, все пакеты Layer 2, приходящие в порт моста, будут направлены в другие порты. Следовательно, набор всех полей “dot1dTpFdbAddress”, меньше тех, что переданы в определенный порт, плюс MAC адрес этого порта, являются коллекцией всех возможных адресатов пакета Layer 2, полученной данным портом.

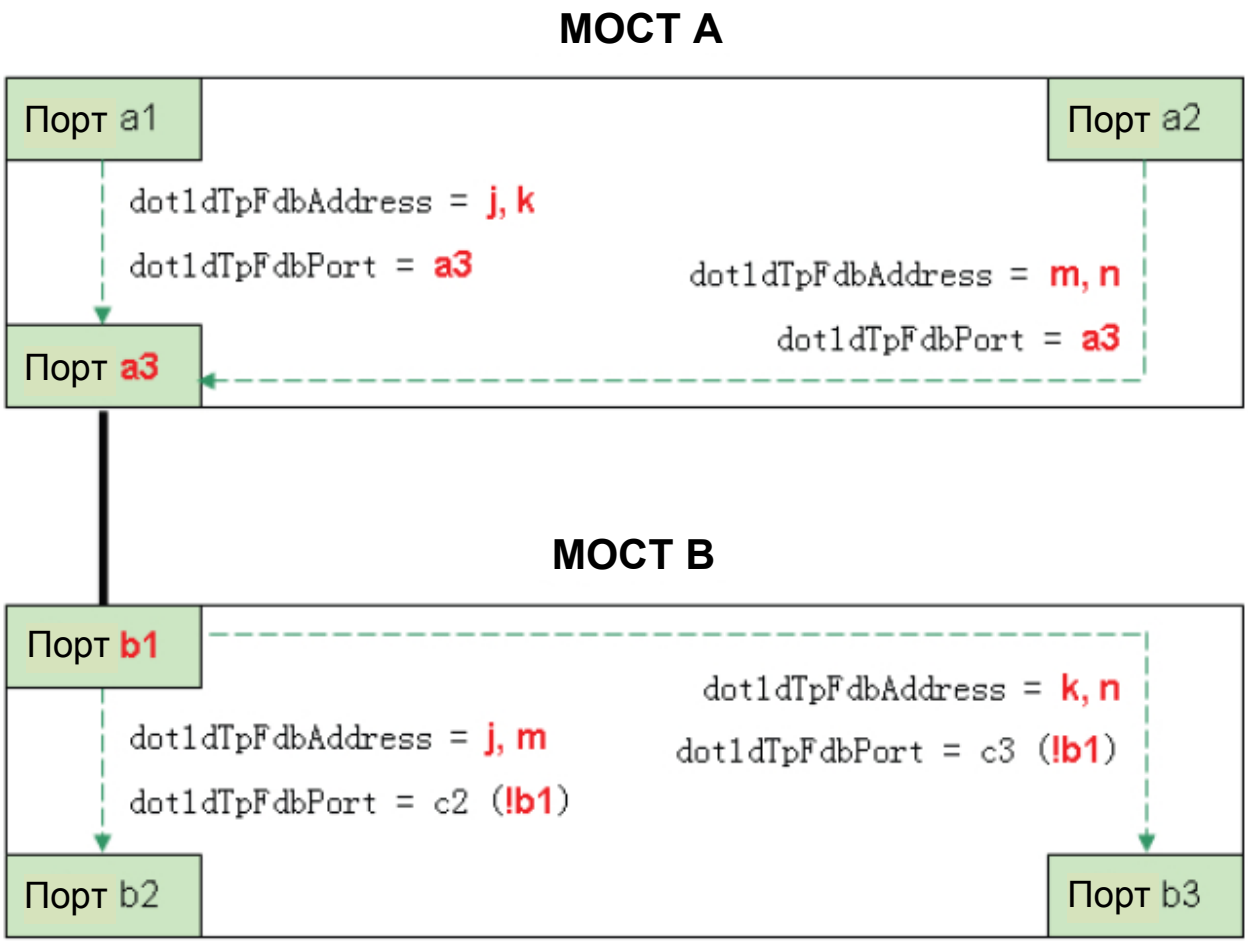

Рис. 13: Теорема соединений Layer2

В физическом межпортовом соединении все пакеты, передаваемые портом, должны приниматься подключенным портом. Поэтому, сравнивая пакеты Layer 2, переданные и полученные двумя портами, можно успешно установить между ними соединение L2-L2.

ТЕОРЕМА СОЕДИНЕНИЯ LAYER 2 (предлагается разработчиком EisStream):

«Если порт x в мосте A соединен с портом y в мосте B, то набор всех полей "dot1dTpFdbAddress" в A с их значением "dot1dTpFdbPort", равным x, меньше любого поля, равного 'ifPhysAddress' в B, должен быть равен набору всех "dot1dTpFdbAddress" с их значениями "dot1dTpFdbPort", неравными y.»

Как видно на Рис. 14, F – это набор всех MAC адресов адресатов пакетов, опуская x, B – набор всех "ifPhysAddress" во всех портах B, а F´ - набор всех "dot1dTpFdbAddress" с их значениями, неравными y:

$$F - B = F'$$

Эта теорема применима ко всем соединениям L2-L2, в т.ч. между портами соединительных линий или мостами со множеством резервов, доступных с основным деревом.

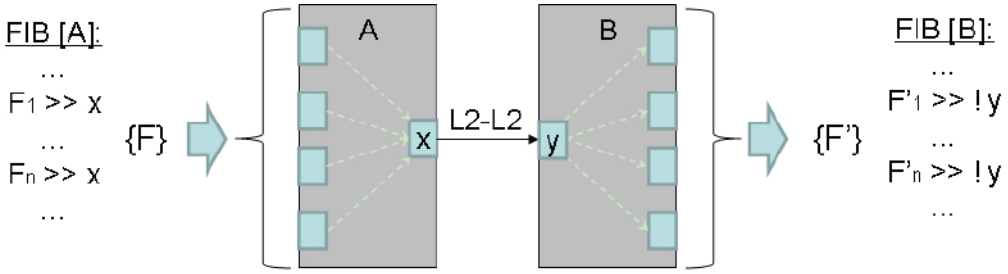

Рис. 14: Мосты, соединенные гирляндной цепью

Из-за сложного характера соединений Layer 2 нереально ожидать, что применение универсального алгоритма даст гарантированную точность и процент успешных попыток. Для надежности в решении практических проблем, таких как присутствие концентратора, неполные MIB данные или некорректные ручные конфигурации, в алгоритм введен элемент нечеткой логики

Вместо того, чтобы искать точное совпадение, алгоритм сначала выберет группу возможных кандидатов по определенному порогу критериев. Затем он применяет к выбранным кандидатам фильтры, чтобы найти самое вероятное совпадение. Поэтому, вместо того, чтобы искать абсолютное совпадение между обеими сторонами уравнения, можно определить порог, выразив его как:

$$(F – B) \cap F' = (F – B) \text{ или } F'$$

Следовательно, теорема также может быть записана так:

«Если порт x в мосте A соединен с портом y в мосте B, то набор всех полей "dot1dTpFdbAddress" в A с их значением "dot1dTpFdbPort", равным x, меньше любого поля, равного 'ifPhysAddress' в B, должен быть ПОДМНОЖЕСТВОМ всех "dot1dTpFdbAddress" с их значениями "dot1dTpFdbPort", неравными y, или наоборот.»

Однако нечеткая логика создает проблемы при определении соединений, включающих 3 или более мостов в гирляндных или кольцевых структурах. В примере на Рис. 15 a3 можно определить алгоритмом нечеткой логики как соединенным с портом b1 или c1. Так происходит потому, что сумма всех возможных адресатов пакетов, кроме a3, является подмножеством всех "dot1dTpFdbAddress" в порте b1, а также их подмножеством в c1.

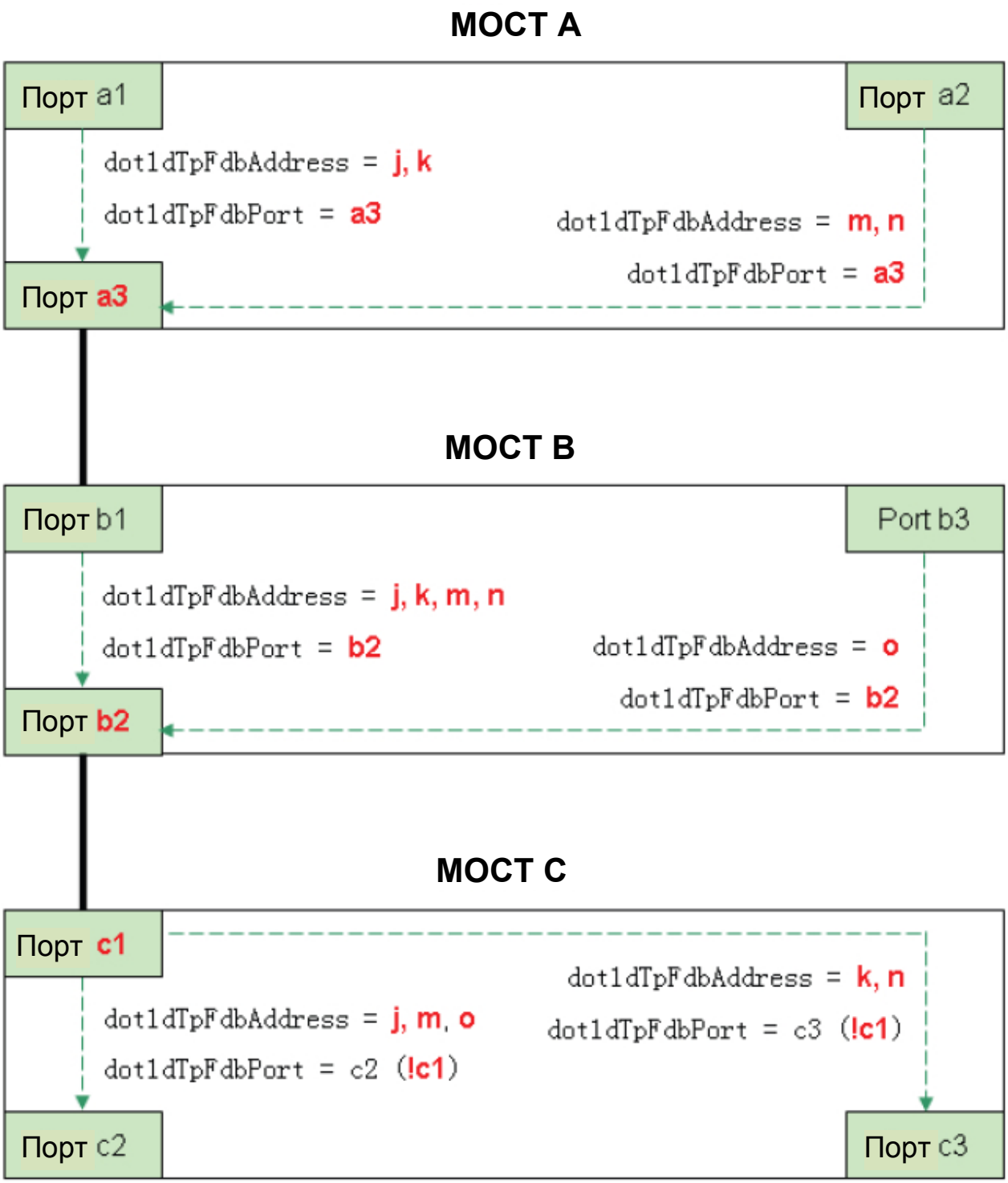

Рис. 15: Мосты, соединенные гирляндной цепью

Эта проблема решается дальнейшим исследованием полей "dot1dTpFdbAddress" в портах, т.к. c1 также передает пакеты на MAC адрес o, которые могут быть отправлены только из b2.

Этот алгоритм нечеткой логики также учитывает присутствие концентраторов. Если концентратор присутствует, один порт в точке разветвления покажется соединенным со множеством хостов, согласно определению алгоритма.

4.2.2.3 Анализ соединений L2-L3

Соединение L2-L3 существует только между портом коммутатора и маршрутизированным портом, где MAC адрес маршрутизированного порта – единственное поле “dot1dTpFdbAddress” в FIB моста со значением “dot1dTpFdbPort”, указывающим на порт коммутатора.

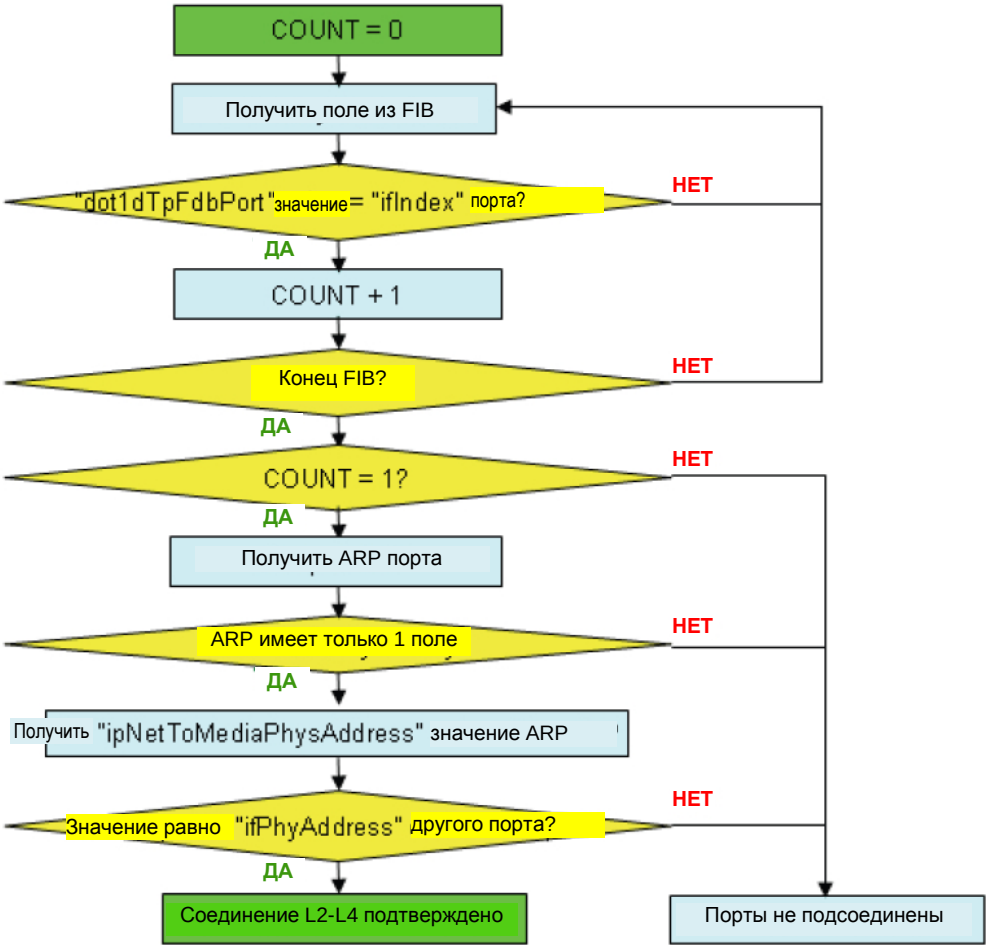

Рис. 16: Алгоритм обнаружения соединений L2-L3

Как видно на Рис. 16, соединение L2-L3 обнаруживается через следующие шаги, которые используются и для обнаружения соединений L3-L2.

1) Найти число полей в FIB моста “dot1dTpFdbPort” равным “ifIndex” порта;
2) Если обнаружено только одно такое поле, проверить число полей в ARP таблице порта;
3) Если обнаружено только одно поле ARP, проверить, совпадает ли “ipNetToMediaPhysAddress” с “ifPhysAddress” другого порта;
4) Если совпадает, то соединение L2-L3 определено.

4.3 End-to-End Physical Path

Отслеживание сквозного физического тракта между двумя хостами в сети – конечная цель построения полной сетевой диаграммы. Однако без правильного использования маршрутной информации Layer 3 по всем маршрутизаторам все равно невозможно точно идентифицировать тракт, где IP пакет должен путешествовать от источника к адресату.

Рис. 17 – пример типичной современной сети из 3 подсетей.

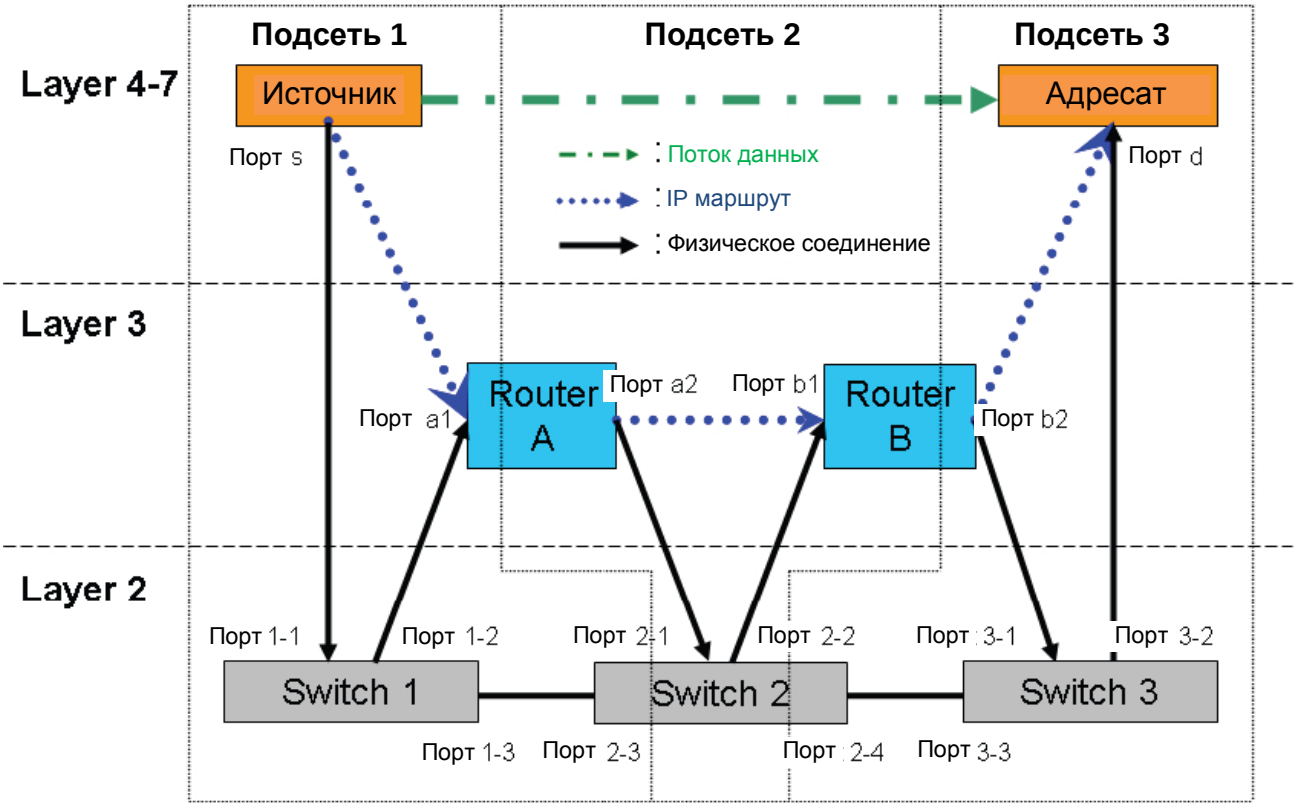

Рис. 17: Пример типичной современной сети из 3 подсетей

Подсети 1 и 2 маршрутизируются через Router A, а подсети 2 и 3 – через Router B.

Мост 2 – магистральный коммутатор с портами, сконфигурированными для всех подсетей, где порт 2-3 соединен с портом 1-3 в Bridge 1 в подсети 1, а порт 2-4 – с портом 3-3 в Bridge 3 в подсети 3.

В смысле физических портов, любой IP пакет от источника в подсети 1 до адресата в подсети 3 будет следовать таким путем:

s >>1-1>1-2 >>a1>a2 >>2-1>2-2 >>b1>b2 >>3-1>3-2 >>d

Где >>" означает физический кабель, а >" – внутреннюю коммутацию или маршрутизацию в устройстве от одного порта к другому.

Однако в простой схеме физической топологии без информации Layer 3 этот путь легко можно истолковать неверно:

s >> 1-1>1-3 >> 2-3>2-4 >> 3-3>3-2 >> d

Чтобы правильно идентифицировать физический тракт для любого сквозного соединения, важно идентифицировать маршруты Layer 3 между источником и адресатом.

4.3.1 Определение маршрутов Layer 3

Изучив установку источника Default Gateway в объекте "ipRouteNextHop", можно легко идентифицировать первый маршрутизатор в маршруте Layer 3. Сетевые номера хостов без этой установки можно вывести из IP адреса в объекте "ipAdEntAddr" и маски подсети в "ipAdEntNetMask". Маршрутизатор можно найти путем сравнения сетевого номера хоста с локально маршрутизированными подсетями каждого маршрутизатора, хранящимися в полях "ipRouteDest" с "ipRouteType", равным 3.

После нахождения маршрутизатора в сети хоста по сетевому номеру адресата или маршруту по умолчанию в его маршрутной таблице "ipRouteTable" или "ipCidrRouteTable" можно найти следующий маршрутизатор. Повторяйте этот шаг, пока не будет найден маршрутизатор в сети адресата.

Важно, чтобы информация интерфейса к следующему маршрутизатору записывалась для каждого маршрутизатора. Для многоуровневых коммутаторов следующий интерфейс может относиться к подсети со множеством коммутаторных портов. В этом случае информация о каждом отдельном коммутаторном порте собирается и хранится вместе с маршрутизатором.

4.3.2 Определение трактов Layer 2

Тракты Layer 2 определяются индивидуально между каждыми двумя соседними маршрутизаторами, а также между маршрутизаторами и хостами в каждом конце маршрута. Определить тракт на схеме между двумя точками относительно просто. Тем не менее, необходим алгоритм, аналогичный Spanning Tree, для программного нахождения тракта.

4.3.2.1 Тракт Layer 2 между маршрутизаторами

В той же подсети, используя информацию физической топологии, отслеживается соединение Layer 2 от всех последующих портов первого маршрутизатора до непосредственных соседей, затем до соседей соседей м так далее, пока не дойдет до следующего маршрутизатора в тракте.

4.3.2.2 Тракт Layer 2 между маршрутизатором и хостом

Тот же алгоритм применяется и для нахождения физического тракта между маршрутизатором и хостом, за исключением того, что начальное устройство всегда является маршрутизатором. Это нужно для учета того факта, что хост всегда можно соединить с маршрутизатором через концентратор, а некоторые хосты могут не поддерживать SNMP.

4.4 Multicast

4.4.1 Группы многоадресной передачи

Извлекая данные “ipMRouteScopeNameString” из всех маршрутизаторов в сети, можно составить полный список групп многоадресной передачи. Адрес многоадресной передачи – это текстовое имя, которое уникально идентифицирует группу многоадресной передачи.

Каждая группа многоадресной передачи содержит источник и ряд получателей. Возможна группа многоадресной передачи без получателя. IP адрес и маска подсети источников хранятся в объектах “ipMRouteSource” и “ipMRouteSourceMask” соответственно.

IP адрес маршрутизатора и маска подсети для Rendezvous Point (RP) группы многоадресной передачи находятся в объектах “ipMRouteRtAddress” и “ipMRouteRtMask” соответственно. IP адрес маршрутизатора, из которого получен определенный многоадресный поток, находится в объекте “ipMRouteUpstreamNeighbor”.

4.4.2 Каналы многоадресной передачи

Для определенной группы многоадресной передачи можно установить полную структуру каналов от источников до всех маршрутизаторов, участвующих в направлении трафика данной группы.

Считав значения “ipMRouteInIfIndex” и “ipMRouteNextHopIfIndex”, можно получить физический порт, из которого принят многоадресный поток, и интерфейс, в который он направлен. Если в маршрутизаторе включено IGMP, передающий интерфейс – это физический порт, а если нет, то подсеть. В случае направления в подсеть многоадресный поток передается на всех коммутаторный порты в этой подсети.

EisStream строит карты многоадресной передачи исключительно на основе информации, собранной из маршрутизаторов многоадресной передачи. Это потому, что структура многоадресной передачи полностью зависит от многоадресных маршрутизаторов, т.к. они договариваются между собой о конечном тракте трафика многоадресной передачи. Мосты, поддерживающие IGMP, которые служат только для функции остановки многоадресных пакетов Layer 2, передаваемых на все порты, пассивно передают пакеты из многоадресных маршрутизаторов в принимающие хосты. Их поведение не соответствует конструкции маршрута многоадресной передачи. Поэтому мосты исключаются EisStream из карты многоадресной передачи. Их существование в трафике можно просто вычислить, найдя сквозной физический тракт между последним многоадресным маршрутизатором и приемным хостом.

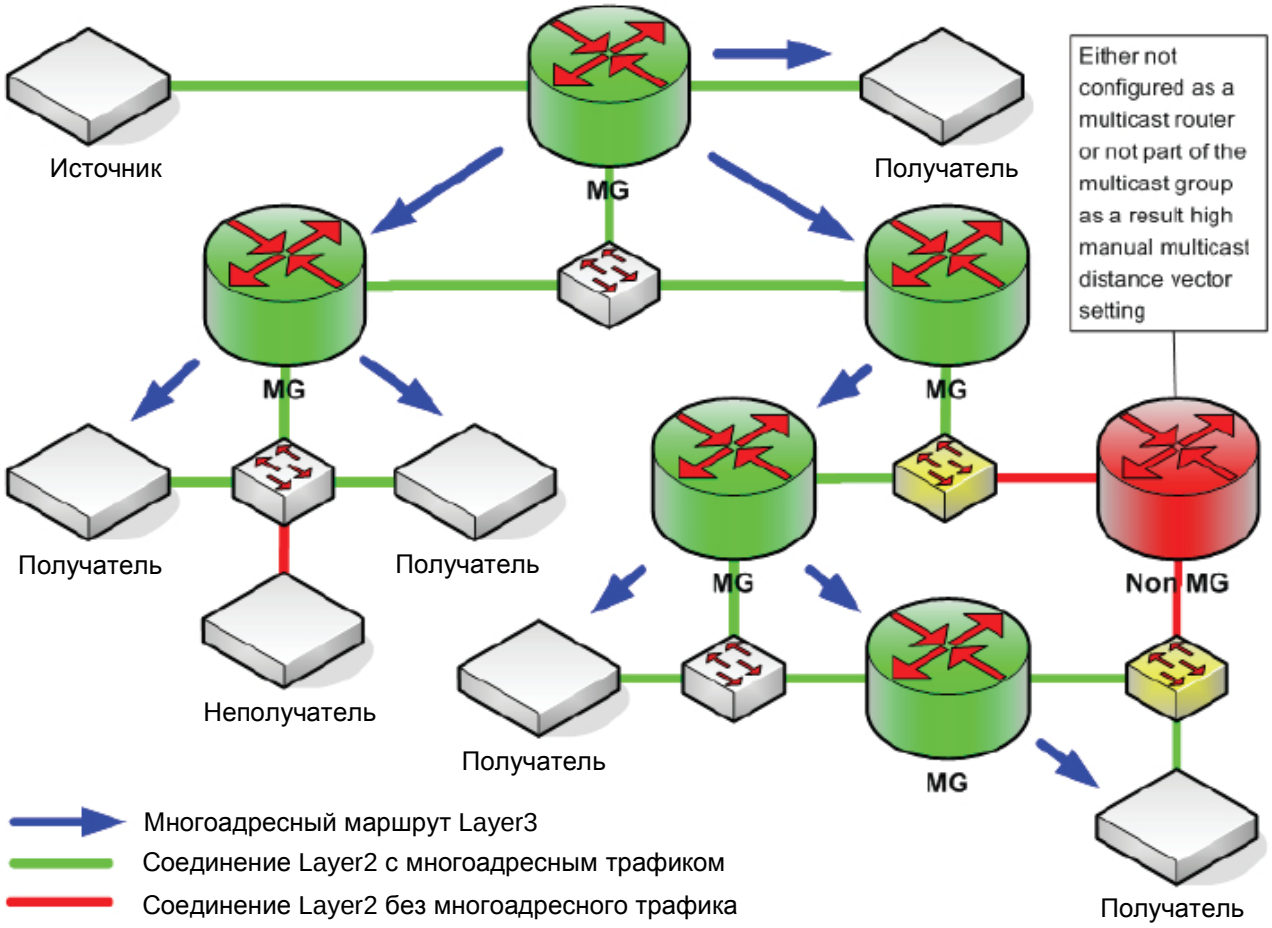

Рис. 18: Карта маршрутов канала многоадресной передачи

В примере на Рис. 18, даже хотя желтые мосты знают более короткий путь через красный маршрутизатор, т.к. красный маршрутизатор не является частью структуры многоадресной передачи, они должны соблюдать маршрут Layer 3 и передавать многоадресные пакеты только в зеленые маршрутизаторы. Поэтому мосты исключаются EisStream из карты многоадресной передачи. Тем не менее, их существование в тракте многоадресного трафика можно просто вычислить, найдя сквозной физический тракт между последним многоадресным маршрутизатором и приемным хостом.

4.4.3 Физический тракт для многоадресного трафика

Используя механизм для обнаружения сквозного физического тракта и информацию о структуре каналов всех групп многоадресной передачи, можно идентифицировать каждый узел, участвующий в передаче определенного многоадресного потока.

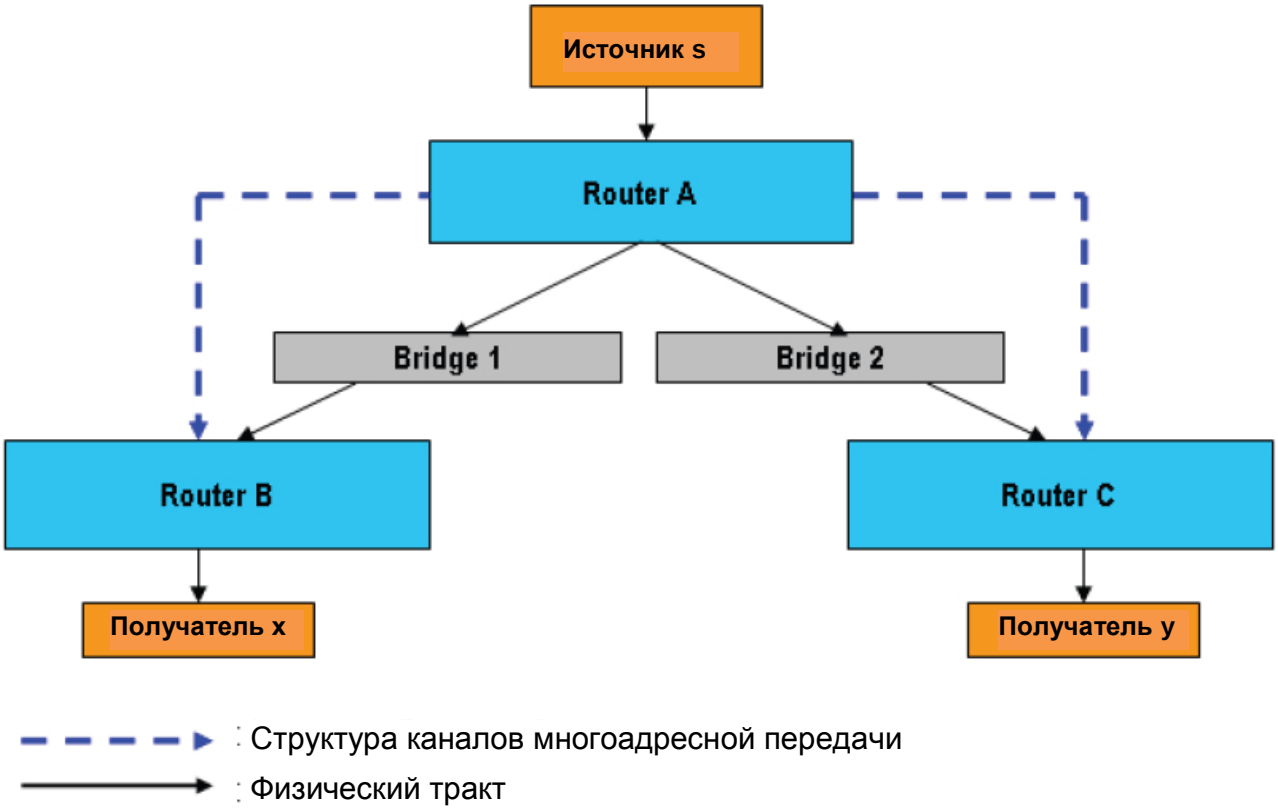

Рис. 19: Физический тракт для многоадресного трафика

Как видно на Рис. 19, физический тракт для соединений между маршрутизаторами, между источником и его RP и между приемниками и многоадресными маршрутизаторами можно определить точно так же, как и для сквозного соединения.

5. Результаты тестов

Программа EisStream была разработана BBC R&D методом Agile Software Development и строго протестирована в сети BBC R&D в каждом цикле итерации.

На Рис. 20 видно, что сеть BBC R&D состоит из одного маршрутизатора и множества мостов и хостов; в сумме более 250.

Каждый прозрачный квадратик на рисунке означает устройство, которое не поддерживает SNMP (EisStream блокируется по времени при попытке контакта с такими устройствами).

Программа также была протестирована напрямую в сети распространения VRT в Бельгии. Результат анализа топологии показан на Рис. 21. Программа смогла обнаружить все маршрутизаторы и мосты в сети. Есть также большое количество хостов, не поддерживающих SNMP в этой сети, в результате чего программа работала более 20 минут, чтобы завершить анализ.

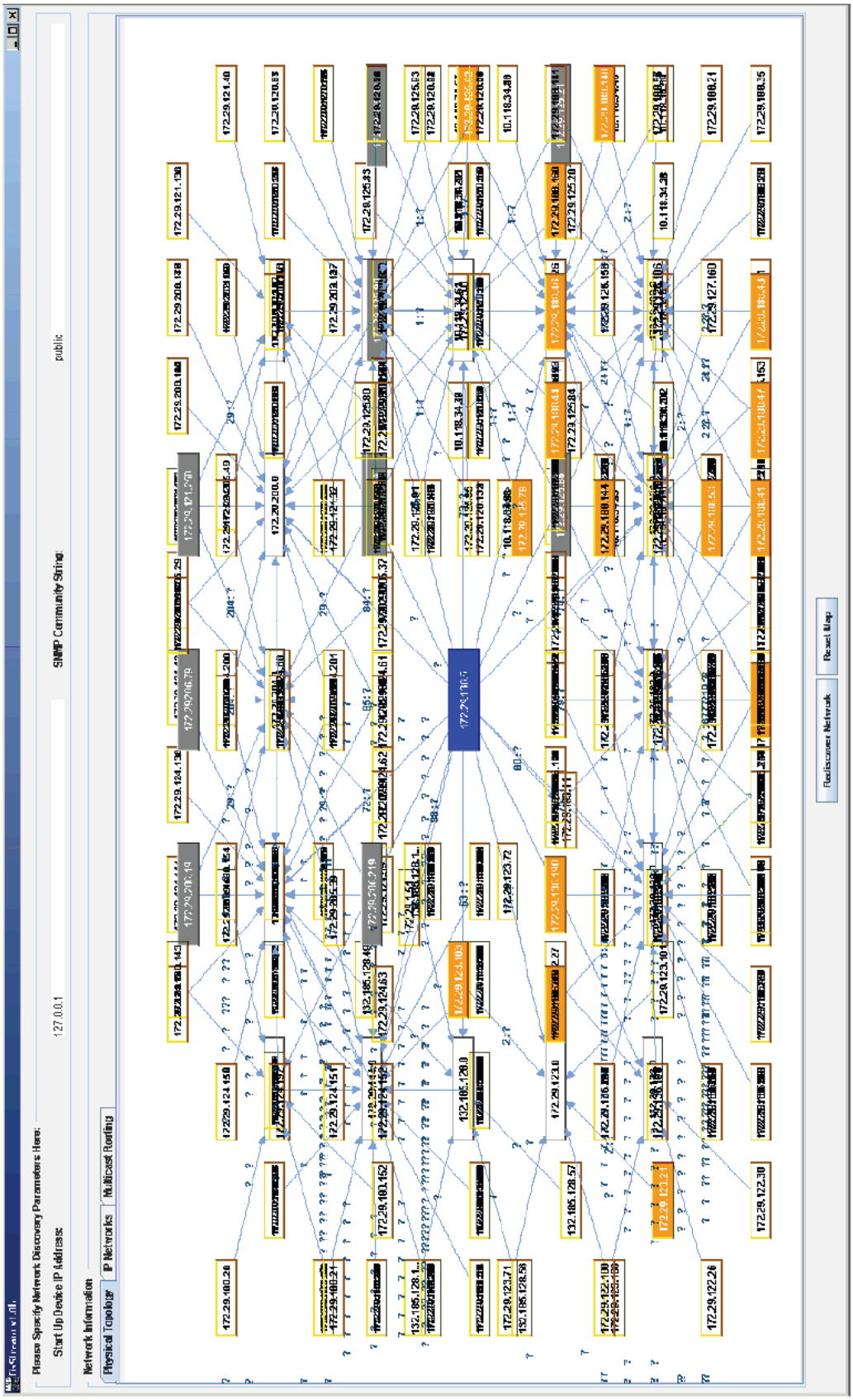

Рис. 20: Анализ EisStream офисной сети BBC R&D

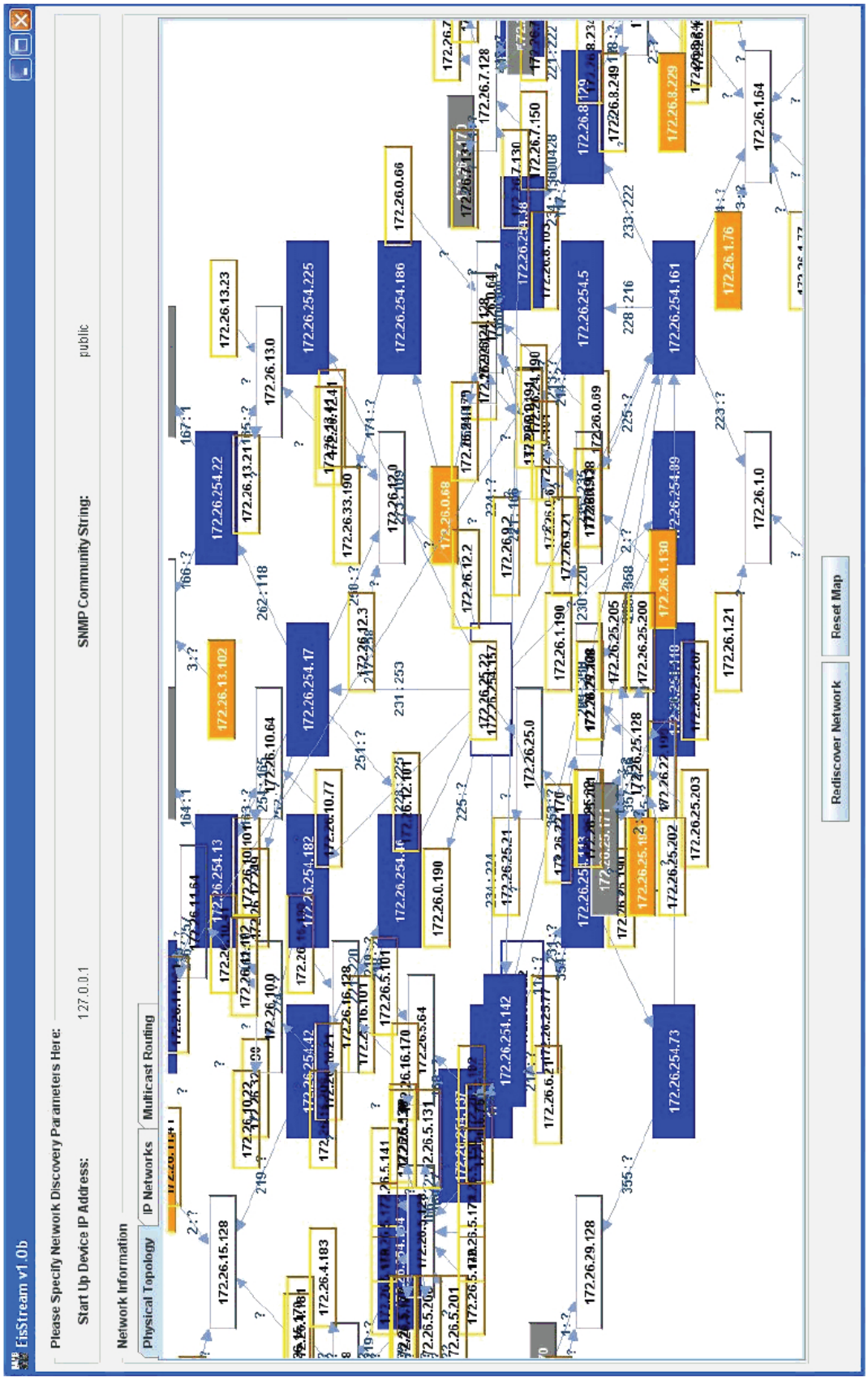

Рис. 21: Анализ EisStream сети распространения VRT

Во всем процессе обнаружения общий сетевой трафик не превышал 500 kbit/s, но использование CPU было 50%, как видно на Рис. 22 и 23. Рекомендуется, чтобы программа работала на CPU с тактовой частотой минимум 1.3 GHz.

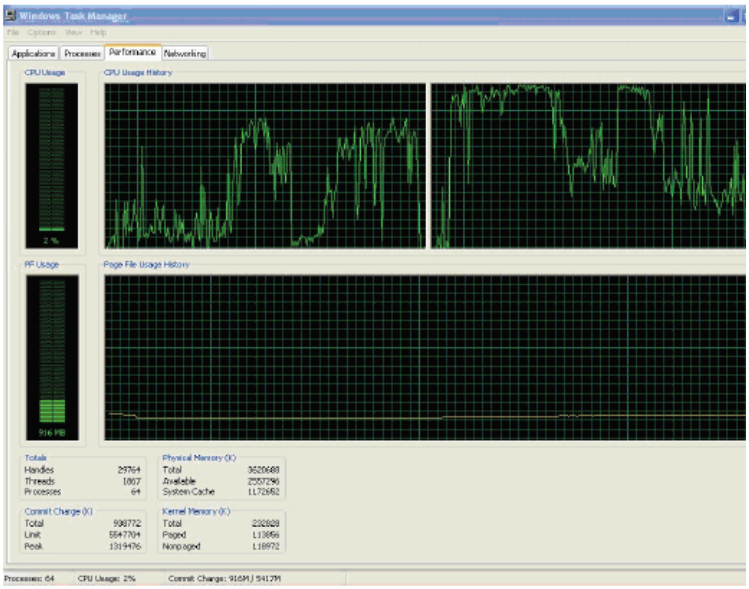

Рис. 22: Нагрузка CPU в PC с EisStream

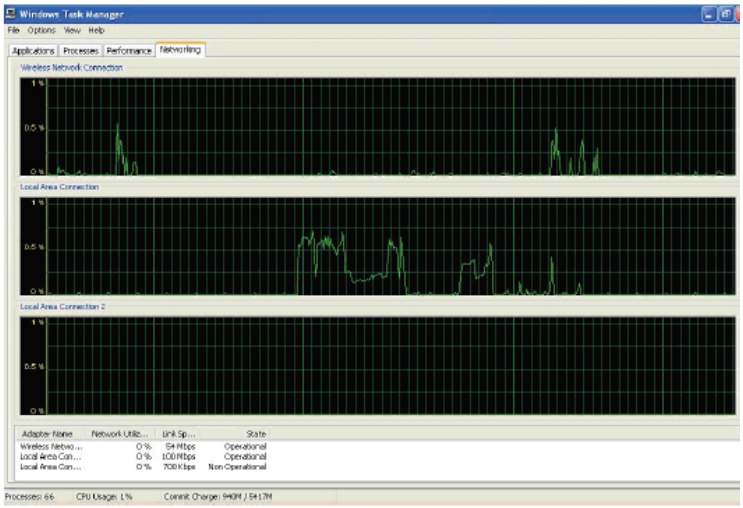

Рис. 23: Использование сети во время анализа EisStream

6. Следующие этапы и дальнейшие разработки

С EBU Tech 3345 и Tech 3346 (настоящим документом), описывающих соответственно новый стандарт сетевых измерений, приспособленный под требования медиа, и универсальную программную платформу, способную контролировать порт любого устройства для медиа потока, теперь имеется интегрированное решение для мониторинга аудиовизуально-ориентированной сети. Следующий шаг – продвижение нового стандарта и инструмента в целях общеотраслевого внедрения главными производителями. Когда новый стандарт MIB будет внедрен повсеместно, программа EisStream сможет достичь конечной цели многоуровневого мониторинга потоков в мультивендорной сети с параметрами, полностью определяемыми медиа.

Функции EisStream потенциально можно внедрить в веб-услуги и реализовать в сценарии удаленного мониторинга, как показано на Рис. 24. API для распределенной или встроенной реализации также могут быть разработаны из существующих пакетов программного обеспечения.

Рис. 24: EisStream как веб-услуга