САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Математико-механический факультет

Кафедра системного программирования

Солозобов Андрей Сергеевич

Динамическая маршрутизация от источника в IP сетях

Дипломная работа

Допущена к защите.

Зав. кафедрой:

д. ф.-м. н., профессор Терехов А.Н.

Научный руководитель:

cт. преп. Баклановский М.В.

Рецензент:

руководитель группы модернизации качества поиска

Яндекс, к.ф.-м.н. Куралёнок И.Е.

Санкт-Петербург

2013

SAINT-PETERSBURG STATE UNIVERSITY

Mathematics & Mechanics Faculty

Software Engineering Chair

Andrei Solozobov

Dynamic loose source routing in IP networks

Graduation Thesis

Admitted for defence.

Head of the chair:

Professor Terekhov A. N.

Scientific supervisor:

Senior Lect. Baklanovsky M. V.

Reviewer:

Search quality modernization team leader,

Yandex, Ph. D. Kuralonok I. E.

Saint-Petersburg

2013

**Содержание**

1. 1. Введение 4

1.1. Актуальность 4

1.2. Предметная область 5

1.3. Обзор существующих решений 8

1.3.1. Статическая балансировка трафика 8

1.3.2. Динамическая балансировка трафика 9

1.3.2.1. Специализированное оборудование 9

1.3.2.2. Программно-конфигурируемая сеть 9

1.3.3. Маршрутизация, определяемая источником 10

1.4. Постановка задачи 11

1. 2. Решение 13

2.1. Описание подхода 13

2.1.1. Алгоритм определения топологии сети 13

2.1.2. Трассировка маршрутов 14

2.1.3. Построение графов связности узлов сети и автономных систем 15

2.1.4. Поиск возможных маршрутов 15

2.1.5. Определение оптимальных маршрутов 16

2.2. Программная реализация 18

2.2.1. Архитектура 18

2.2.2. Выбор способа трассировки маршрутов 19

2.2.3. Параметры трассировки маршрутов 19

2.2.4. IP – ASN разыменование 22

2.2.5. Настройка сетевых соединений 23

1. 3. Эксперименты 25

3.1. Первый и второй эксперименты 25

3.1.1. Первый эксперимент 26

3.1.2. Второй эксперимент 27

3.2. Третий эксперимент 28

1. 4. Заключение 31

4.1. Результаты 31

4.2. Развитие работы 31

**Список литературы**

# 

# Введение

## Актуальность

Дефицит пропускной способности каналов связи между прикладными процессами, разнесёнными в пространстве на значительные расстояния, является серьёзной проблемой. Задача повышения эффективности использования ресурса пропускной способности из-за высокой стоимости создания каналов связи в наше время остаётся актуальной.

Причиной дефицита пропускной способности канала обмена данными могут служить как фактическое исчерпание пропускной способности физического и канального уровней, так и нестабильность работы каналов связи. При этом пользователь (процесс), столкнувшийся с этой проблемой, как правило, оказывается не в состоянии в одиночку решить сложившуюся проблему. Для определения её причин необходимо обладать комплексным взглядом на принципы построения сети и её устройство, а также знанием принципов и протоколики её функционирования, топологии и характеристик связей.

Даже локализованный и классифицированный источник ограничения пропускной способности может оказаться физически недосягаемым, в том числе по форс-мажорным или иным обстоятельствам непреодолимой силы. Поэтому решение проблемы часто требует административного и финансового ресурса, согласования со сторонними лицами и организациями, отчего затягивается, а иногда и останавливается, будучи осложнённым бюрократическими, конъюнктурными и другими причинами.

Тем не менее, с дефицитом пропускной способности каналов обмена данными сети и системы сталкиваются регулярно, поэтому решение этой проблемы актуально и востребовано.

## Предметная область

Множество компьютерных сетей по всему миру объединены протоколом IP [1][2] во всемирную сеть Интернет. В наши дни она обеспечивает основу функционирования множества других систем передачи данных, ежедневно используемых миллионами людей.

Обмен данными в IP-сети основывается на наличии у каждого участника сети уникального номера - IP-адреса, по которому производится адресация и доставка данных. За доставку данных отвечают специальные маршрутизирующие узлы сети. Каждый из них руководствуется определённым набором правил маршрутизации, по которым он пересылает данные в направлении адресата. В результате последовательного применения таких правил и пересылок по цепочке маршрутизирующих узлов данные доходят до адресата. Для этого правила маршрутизации специальным образом настраивают вручную или же формируют по специальным алгоритмам маршрутизации.

Каждый маршрутизирующий узел хранит правила маршрутизации в специальной структуре, называемой таблицей маршрутизации. Она описывает соответствие групп IP-адресов ведущим к ним от маршрутизатора сетевым соединениям. Каждое из этих правил имеет приоритет. Для пересылки данных адресату маршрутизатор среди всех подходящих использует правило с наивысшим приоритетом. Это правило используется до тех пор, пока не теряет приоритет или не исключается из-за потери связи в соответствующем сетевом соединении.

В Интернете выделяют логические объединения узлов, называемые автономными системами AS (Autonomous System) [3]. Каждая из них реализует внутри себя определённую политику маршрутизации и связана с другими автономными системами посредством выделенного набора узлов, называемых граничными маршрутизаторами (шлюзами). Адресация и маршрутизация на уровне автономных систем определяется протоколом BGP и реализуется граничными маршрутизаторами, работа которых также основывается на таблицах маршрутизации.

При таком способе организации маршрутизации в каждый момент времени передача данных от одного фиксированного участника сети другому происходит физически лишь по одной последовательности маршрутизаторов и каналов передачи данных.

Каждый физический канал передачи данных характеризуется своей канальной пропускной способностью – максимальным количеством информации, передаваемым по нему в единицу времени. По различным причинам передаваемые в канале данные могут быть потеряны или доставлены адресату в изменённом виде. Для обеспечения гарантированной доставки, сохранения порядка и целостности передаваемых через физический канал данных прикладные процессы устанавливают друг с другом соединение по специальным протоколам транспортного уровня модели OSI, например, TCP. Каждое соединение характеризуется своей информационной пропускной способностью – максимальным количеством полезной информации, передаваемым в соединении в единицу времени. Из-за использования одного физического канала несколькими прикладными процессами и передачи через него служебной информации, предназначенной для поддержания соединения, информационная пропускная способность всегда ниже канальной пропускной способности.

В тот момент, когда оказывается, что канал перегружен, то есть спрос на передачу данных превышает пропускную способность канала, начинает нарастать очередь данных, ожидающих передачи. После достижения максимальной длины этой очереди поступающие данные начинают отвергаться. Перегрузка канала передачи данных приводит к тому, что отправивший данные процесс вовремя не получает подтверждения о доставке адресату и переотправляет их, вызывая тем самым многократное увеличение нагрузки на канал. Таким образом при сохранении неизменной канальной пропускной способности, информационная пропускная способность передачи данных при наступлении перегрузки существенно падает.

## Обзор существующих решений

Проблема дефицита пропускной способности канала обычно решается обновлением программной и аппаратной его составляющих с целью увеличения пропускной способности, или введением в эксплуатацию дополнительного канала. Второй способ решения проблемы повышает надёжность работы сети, так как дополнительный канал будет служить резервным источником связи на случай выхода из строя основного, но в таком случае сетевым администраторам необходимо будет определить правила распределения трафика (балансировки) по каналам и соответствующим образом настроить оборудование.

### Статическая балансировка трафика

Подход статического определения настроек балансировки является дёшевым, так как не требует затрат на специализированное программное обеспечение и оборудование.

Для облегчения работы сетевыми администраторами строятся специальные карты сети, схемы расположения узлов, маркеруются кабели. Тем не менее, топологию и рабочие нагрузки в сети даже из сотни машин нелегко представить и правильно сбалансировать. Всегда остаётся место ошибке и человеческому фактору. Кроме того, топология сети и распределение в ней нагрузки со временем меняются, что приводит к смещению ранее выверенных балансов и необходимости перенастройки оборудования.

Отсутствие адаптивности данного подхода часто делает его непригодным.

### Динамическая балансировка трафика

Подход динамической балансировки тафика является более сложным и дорогим по сравнению со статическим. Он требует использования специальных протоколов и обрудования, способного работать по этим протоколам.

### Специализированное оборудование

Существует группа протоколов, позволящих отдельным маршрутизаторам балансировать трафик между соседними узлами с одинаковыми метриками. В эту группу входят протоколы EIGRP, и CEF разработанные фирмой CISCO и поддерживаемые маршрутизаторами её последних моделей. Такие протоколы хранят карту коммутации узлов FIB (Forwarding Information Base), которая позволяет им для каждого адресата определять несколько следующих узлов пересылки и, используя различные политики, балансировать трафик между ними.

Решения такого рода балансируют трафик лишь локально: в рамках одного или нескольких узлов сети, в которой работа организована по специальному протоколу.

### Программно-конфигурируемая сеть

Программно-конфигурируемые сети SDN (Software-Defined Network) − относительно недавно появившаяся концепция организации сетей передачи данных, выделяющая в структуре сети центральный узел, который определяет топологию сети, формирует логические сети и правила маршрутизации, устанавливает приоритеты трафика и удалённо конфигурирует все остальные узлы сети, которые занимаются лишь передачей трафика, и, соответственно, могут быть более простыми в устройстве и дешевыми.

Такой подход даёт сетевым администраторам единый интерфейс настройки сети, позволяющий в автоматическом режиме программно оптимизировать работу локальной сети и формировать сетевые соединения с необходимыми параметрами. С другой стороны, центральный узел становится единой точкой отказа всей сети, без которой работа сети невозможна, о его резервировании нужно заботиться отдельно.

Реализацией такой концепции является протокол OpenFlow, который уже поддерживается многими производителями сетевого оборудования, такими, как CISCO, HP, IBM. Хотя текущая версия спецификации протокола и вызывает нарекания у сетевых администраторов, к разработке протокола подключились такие компании как Google и Facebook.

### Маршрутизация, определяемая источником

Существует подход к построению сетей, в котором функцию маршрутизатора выполняет сам отправитель на основе информации о топологии сети, получаемой от других узлов. Его воплощением является протокол WRAP [4], работающий поверх протокола IP между транспортным и сетевым уровнями модели OSI.

Заголовок WRAP-пакета несёт в себе записанную отправителем последовательность IP-адресов, через которые пакет уже прошел и ещё должен будет пройти. Этот подход позволяет пользователям выбирать каналы движения трафика и тем самым достигать необходимых характеристик виртуального канала связи, а сетевым администраторам реализовывать более сложные механизмы фильтрации трафика.

На данный момент этот протокол не реализован ни одним из известных производителей сетевого оборудования и существует только как концепция.

## Постановка задачи

При проектировании современных IP-сети принимаются меры для повышения их отказоустойчивости. Используются приёмы создания запасных каналов связи: дублирование связей, топологические кольца, утолщённые деревья. Если представить такую сеть в виде графа связности узлов (см. Приложение 1), то в нём обнаружится множество циклов. Это означает, что во многих случаях данные от источника к получателю могут быть переданы по альтернативному маршруту, минимально пересекающемуся по узлам и рёбрам графа с определённым сетью стандартным маршрутом передачи данных.

В протоколе IP заложена возможность передачи данных по нестандартным маршрутам. Для этого в заголовке IP-пакета можно указать специальную опцию маршрутизации от источника в виде списка IP-адресов маршрутизаторов сети, через которые данный пакет должен пересылаться при доставке адресату.

Одновременная передача данных по стандартному и альтернативным маршрутам позволит снизить нагрузку на стандартный маршрут передачи данных и повысить пропускную способность соединения с адресатом. Особенно выигрышным, в смысле повышения пропускной способности соединения, такой подход окажется в случае перегрузки каналов стандартного маршрута передачи данных.

Для реализации описанного подхода необходимо обладать знанием о наличии связности маршрутизаторов в используемых сетях (топологии сетей), уметь находить в графах связей альтернативные маршруты передачи данных, распределять данные между найденными маршрутами и передавать данные по нестандартным, в смысле правил маршрутизации, маршрутам.

В рамках дипломной работы решение задачи повышения пропускной способности соединений с заданными пользователями путём одновременного использования нескольких каналов передачи данных было разбито на подзадачи:

1. Разработать алгоритм определения связности узлов IP-сети в заданном направлении.
2. Разработать алгоритм поиска и выбора альтернативных маршрутов передачи данных в заданном направлении.
3. Создать программную реализацию способа динамического определения маршрутов движения трафика на основе разработанных алгоритмов и опций маршрутизации от источника протокола IP.
4. Провести эксперимент по оценке эффективности передачи данных с использованием разработанной программной реализации.

# Решение

## Описание подхода

Повышение пропускной способности сетевого соединения между двумя процессами, передающими данные по перегруженному каналу, достигается за счёт переноса части сетевой нагрузки с основного канала на обнаруженные в сети дополнительные.

Во время работы программы в направлении каждого целевого узла сети, с которыми ведётся обмен данными, происходит определение и сохранение топологии сети в виде графов. На графах выполняется поиск возможных маршрутов передачи данных. Среди обнаруженных маршрутов выбирается заранее определённое количество оптимальных, поочерёдно используемых для обмена данными с целевым узлом сети.

### Алгоритм определения топологии сети

Заголовок IP-пакета имеет поле времени жизни TTL (Time To Live), хранящее максимальное допустимое количество пересылок данного пакета при доставке адресату. Каждый узел, получающий IP-пакет, уменьшает значение этого поля на единицу и пересылает пакет дальше. Если же время жизни становится равным нулю, то такой пакет отвергается, и, как правило, отправителю высылается ICMP [5] ответ об истечении времени жизни передаваемого пакета.

Для определения связности узлов сети используется так называемый подход трассировки маршрутов, основывающийся на отправке целевым узлам IP-пакетов с различными значениями времени жизни.

### Трассировка маршрутов

Алгоритм отправляет целевому узлу IP-пакеты, последовательно увеличивая их время жизни (начиная с единицы). Отправив пакет со временем жизни N, алгоритм ожидает ICMP-ответа от N-го маршрутизатора в цепочке, передающей исходный пакет. По последовательности полученных ответов он строит эмпирическое представление о последовательности маршрутизаторов доставляющих пакет целевому узлу.

К сожалению, ICMP-ответы приходят не всегда, что мешает процессу трассировки. При разработке алгоритма трассировки были выделены три характерных типа возникающих нештатных ситуаций, связанных с отсутствием ICMP-ответа, и способов борьбы с ними:

* **Ответный ICMP-пакет был отправлен, но по неизвестным причинам не дошел до нас**

На этот случай пакет переотправляется повторно заранее определённое фиксированное количество раз.

* **Промежуточный узел не отвечает или фильтрует запросы, приходящие на указанный порт**

Отправляется второй пакет на другой порт. При получении непрерывной последовательности из заранее определённого фиксированного количества не ответивших хостов трассировка маршрута завершается.

* **Пакет начинает ходить по кругу**

Трассировка маршрута заканчивается, последовательность выписывается в лог и не обрабатывается.

Из-за задержек сети возможна ситуация получения ICMP-ответов не в хронологическом порядке их отправления, что в свою очередь ведёт к построению неверного маршрута. Для исключения возможности таких событий все пакеты отправляются последовательно с заранее определённой фиксированной задержкой.

Также алгоритм трассировки позволяет определять маршруты доставки данных целевым узлам через произвольные промежуточные узлы сети путём добавления к отправляемым IP-пакетам опций маршрутизации от источника.

### Построение графов связности узлов сети и автономных систем

Алгоритм строит графы связности узлов (IP-граф) и автономных систем (BGP-граф) сети на основе результатов трассировки целевых узлов через сторонние узлы сети. В зависимости от настройки, алгоритм может выполнять поиск маршрутов как внутри автономной системы, когда сторонние узлы выбираются случайно из диапазона IP-адресов автономной системы отправителя, так и вне её, когда сторонние узлы выбираются случайно из диапазона допустимых IP-адресов.

Из получаемых при трассировке маршрутов исключаются не ответившие узлы, а оставшиеся последовательности IP-адресов добавляются в IP-граф. Затем происходит разыменование IP-адресов в номера автономных систем и пополнение BGP-графа.

Для хранения каждого из IP и BGP графов связности узлов сети используются хэш-таблицы, сопоставляющие каждой вершине графа список смежных с ней, позволяющие быстро проводить проверку наличия вершины в графе и добавлять в него новые связи.

### Поиск возможных маршрутов

Алгоритм поиска маршрутов опеделяет на графе набор путей, связывающих две заданные вершины (отправителя и получателя). Он представляет собой модификацию алгоритма двунаправленного поиска и выполняется в два этапа.

На первом этапе для отправителя и получателя итеративно строятся множества вершин, достижимых за N переходов по рёбрам графа. Изначально N = 0 и каждое из двух множеств состоит из одной вершины: отправителя или получателя. Затем на каждой итерации N увеличивается на единицу и множество пополняется вершинами, смежными с уже содержащимися в нём. Для каждой добавляемой в множество на шаге N+1 вершины сохраняется предшествующая смежная с ней вершина из множества шага N. Увеличение N происходит до тех пор, пока множества отправителя и получателя не перестанут расти или не начнут пересекаться больше, чем по заранее определённому количеству элементов.

На втором этапе, на основе сохранённого отображения предшествования вершин, для каждой вершины из пересечения множеств происходит восстановление пути, по которому она связана с отправителем и получателем.

В зависимости от настройки, алгоритм может выполнять как простой поиск путей на всём IP-графе, так и поиск путей на ограниченном наборе допустимых вершин. В качестве такого набора используется объединение диапазонов IP-адресов автономных систем, входящих в найденные при поиске на BGP-графе путей между автономными системами отправителя и получателя.

### Определение оптимальных маршрутов

Для передачи данных из множества найденных алгоритмом поиска возможных маршрутов требуется выбрать определённое количество лучших. Необходимо, чтобы выбранные маршруты, по возможности, не имели общих каналов передачи данных, чтобы минимизировать возможность их перегрузки.

Процесс перебора и оценки оптимальности всех возможных подмножеств множества найденных маршрутов довольно трудоёмок для частого выполнения во время работы программы. Чтобы его избежать, было решено составлять набор маршрутов из стандартного, определяемого сетью, и альтернативных, по возможности, минимально пересекающихся со стандартным, маршрутов. Также при выборе маршрутов было решено отдавать предпочтение более коротким маршрутам, как потенциально более производительным.

При выборе альтернативных маршрутов производилось их упорядочение по значению специальной функции характеристики, отражающей долю совпавших рёбер альтернативного и стандартного маршрутов и длину альтернативного маршрута. Для её вычисления определялось количество совпавших рёбер С, количество уникальных рёбер у основного маршрута A и количество уникальных рёбер у альтернативного маршрута B. Характеристика альтернативного маршрута вычислялась по формуле 1.

Формула 1 расчёта характеристики маршрута

Дробь характеризует долю рёбер стандартного маршрута, присутствующих в альтернативном. Чем меньше это значение, тем больше отличается альтернативный маршрут от стандартного и тем он лучше. Член является длиной альтернативного маршрута. Чем она меньше, тем короче маршрут.

Среди найденных маршрутов выбиралось заранее определённое фиксированное число маршрутов с наименьшим значением характеристики, которые затем использовались для передачи данных.

## Программная реализация

Программная реализация данного подхода была выполнена на языке C++ и встроена в библиотеку с открытыми исходными кодами ASIO, входящую в набор С++ библиотек BOOST. Исходные коды доступны для скачивания из SVN репозитория[[1]](#footnote-1).

### Архитектура

При реализации данного подхода исходный код было решено разделить на три модуля:

### Модуль определения топологии сети

* определяет связность узлов сети на уровнях протоколов IP и BGP, используя алгоритм трассировки маршрутов через промежуточные узлы сети
* строит представление топологии сети в виде графов связности узлов (уровень протокола IP) и автономных систем (уровень протокола BGP)

### Модуль анализа топологии сети и построения маршрутов

* по графам связности узлов сети с помощью разработанного алгоритма определяет возможные наборы маршрутов передачи данных в заданных направлениях
* среди найденных определяет оптимальные наборы маршрутов передачи данных в заданных направлениях

### Модуль передачи данных IP-соединений

* используя опции маршрутизации от источника, конфигурирует соединения с целевыми узлами для равномерного распределения передаваемых данных между стандартным и альтернативными маршрутами

### Выбор способа трассировки маршрутов

Известные способы сканирования узлов сети хорошо описаны в статье “The Art of Port Scanning”[6]. Из них три способа пригодны для реализации трассировки маршрутов:

* отправка ICMP-запросов типа Echo Request – на большинстве современных операционных требует привилегий суперпользователя;
* отправка TCP [7] SYN-пакетов, инициирующих установку соединения по протоколу TCP – для отправки такого запроса требуется использование “сырых” сокетов, для создания которых необходимы привилегии суперпользователя;
* отправка UDP [8] пакетов – требует выбора порта назначения.

Был выбран способ трассировки UDP-пакетами, как наиболее простой, не требующий привилегий суперпользователя и создания соединений. При формировании пакетов порты назначения выбирались последовательно из заранее определённого списка.

### Параметры трассировки маршрутов

Для определения параметров, оптимальных с точки зрения полноты получаемых данных и производительности процесса трассировки маршрутов, была проведена экспериментальная трассировка маршрутов до случайных IP-адресов Интернета.

ICMP-ответы на трассировочные UDP-пакеты ожидались не более двух секунд. В случае отсутствия ответа пакет переотправлялся до пяти раз. После последовательности из семи подряд идущих не ответивших узлов трассировка данного IP-адреса прекращалась. Всего в ходе трассировки было обнаружено 2090 узлов.

Параметр максимального времени ожидания ответа был выбран равным 800 миллисекундам. В него в среднем уложилось 99% узлов. График распределения количества узлов по времени ответа, в которое они уложились, приведён на рисунке 1.

Рисунок 1 Распределение количества узлов по времени ответа, в которое они уложились

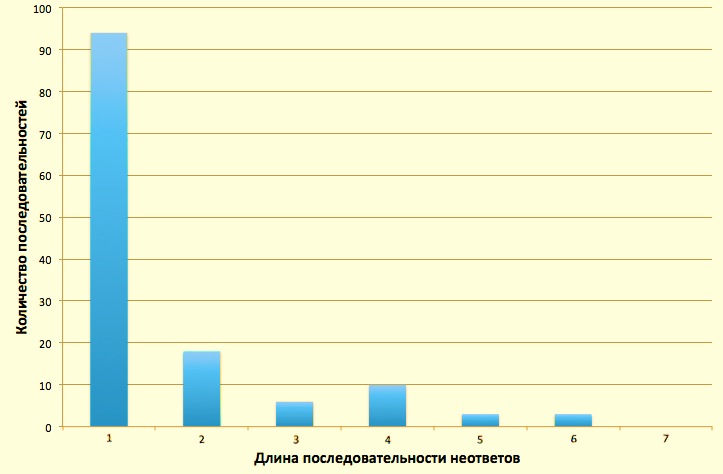
В полученных трассах были обнаружены 134 различные последовательности узлов сети, намеренно не отвечавших на запросы, завершавшиеся узлом сети, отправлявшим ответ. Распределение количества последовательностей в зависимости от их длины приведено на рисунке 2.

Рисунок 2 Распределение количества последовательностей “неответов”по их длине

Значение длины последовательности не отвечающих узлов, при котором трассировка останавливалась, было выбрано равным 4. В него уложились 95% всех полученных последовательностей “неответов”.

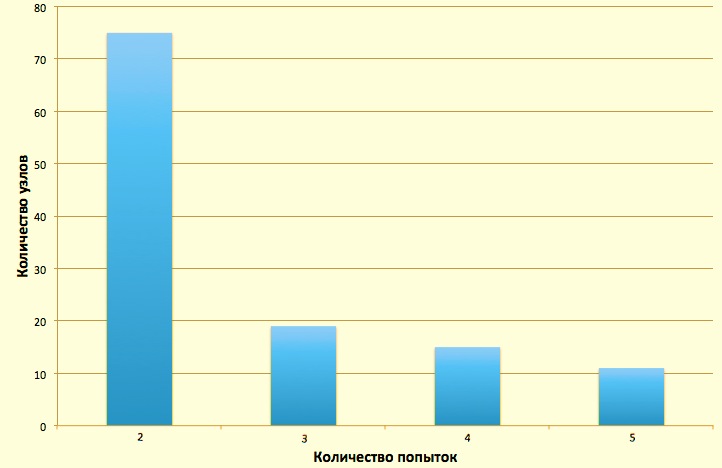
Было обнаружено 120 нестабильно отвечающих узлов (от которых ответ не был получен с первого раза) − 6% от общего количества. Трассировка каждого из этих узлов была отдельно проведена ещё десять раз до пяти попыток отправки пакета. Распределение количества нестабильно отвечающих узлов по количеству попыток, за которое они в среднем ответили, приведено на рисунке 3.

Рисунок 3 Распределение количества нестабильно отвечающих узлов по среднему количеству попыток отправки трассировочного запроса до получения ответа

Параметр количества попыток переотправки пакета не ответившему узлу был выбран равным двум. За две попытки в среднем ответили 96,5% всех узлов.

### IP – ASN разыменование

На основе обновляющегося графа IP-связей строится граф связности автономных систем. Для его построения необходимо уметь делать прямое разыменование IP-адресов в номера автономных систем. Также для поиска возможных промежуточных узлов и передачи данных через определённую автономную систему необходимо уметь делать и обратное разыменование.

Каждый региональный интернет-регистратор хранит базу принадлежности выделенных им IP-адресов автономным системам, которую предоставляет в открытый доступ в Интернете. Эти разрозненные базы очень удачно сведены на сайте APNIC RIR[[2]](#footnote-2).

Во время слияния баз данных выяснилось, что формат их представления неоднозначен. Встречаются коллизии пересечения диапазонов IP-адресов автономных систем, например, AS13703 - 216.88.0.0/20 и AS3561 - 216.88.0.0/14, для разрешения которых соответствующим регистраторам приходится делать дополнительные whois запросы. По этой причине наиболее удобным оказалось использование уже собранной, скорректированной и обновляемой раз в три месяца базы IP – ASN соответствий GeoLite ASN[[3]](#footnote-3).

Для быстрого поиска по базе был построен равномерно распределённый по диапазону IP-адресов индекс на 1024 ключа. В качестве ключа брался префикс из первых 10 бит IP-адреса.

### Настройка сетевых соединений

Работа модуля настройки сетевых соединений основывается на таблице маршрутов, которая хранит для каждого узла сети, с которым ведётся обмен данными, соответствующий ему набор найденных оптимальных маршрутов. Записи этой таблицы периодически обновляются по результатам работы алгоритма поиска оптимальных маршрутов на графе связности сети.

Для работы с сетью в библиотеке ASIO используются стандартные сокеты операционной системы Unix. Настройка различных параметров работы стека сетевых протоколов, используемых при отправки данных через сокет производится с помощью стандартной функции setsockopt(), в том числе позволяющей устанавлиать опции маршрутизации от источника протокола IP.

При вызове пользователем метода передачи данных производится поиск связанного с используемым сокетом IP-адреса в таблице маршрутов. Если в ней содержатся несколько возможных маршрутов, то все они поочерёдно по кругу начинают использоваться для передачи адресату блоков данных. Если же записи в таблице отсутствуют, то для передачи данных используется стандартный маршрут, а IP-адрес сохраняется для дальнейшего поиска и определения альтернативных маршрутов.

# Эксперименты

В рамках экспериментов между двумя целевыми узлами (отправителем и получателем данных) хотелось получить избыточную в смысле количества маршрутов обмена данными сеть передачи данных с перегруженным основным маршрутом и возможностью передачи по сети IP-пакетов с опцией маршрутизации от источника.

## Первый и второй эксперименты

В экспериментах участвовали четыре одинаковых компьютера. В каждом из них работали два сетевых интерфейса: Ethernet и WiFi сетевые карты. Из компьютеров была построена сеть в форме четырёхугольника, в котором связи на противоположных ребрах организованы посредством кабельных Ethernet соединений и WiFi соединений точка-точка.

Принимающим и передающим данные узлами были выбраны компьютеры, имеющие прямое WiFi соединение, остальные два компьютера были маршрутизирующими узлами сети и образовывали альтернативный прямому соединению канал передачи данных от отправителя к получателю.

Рисунок 4 Схема построения сети для первого и второго экспериментов

Контроль прохождения обработки передаваемых пакетов на компьютерах производился с помощью анализатора трафика Wireshark.

### Первый эксперимент

В первом эксперименте все компьютеры работали под управлением операционной системы Windows 7 Enterprise SP1 32-bit v6.1 Build 7601. На маршрутизирующих компьютерах была запущена служба “маршрутизации и удалённого доступа”, в реестре были отредактированы ключи:

* маршрутизация поступающих пакетов HKEY\_LOCAL\_MACHINE/SYSTEM/CurrentControlSet/services/Tcpip/Parameters/IPEnableRouter = 1
* обработка маршрутизации от источника в IP-пакетах HKEY\_LOCAL\_MACHINE/SYSTEM/CurrentControlSet/services/Tcpip/Parameters/DisableIPSourceRouting = 0

После перезагрузки с такими настройками операционная система начинала корректно обрабатывать опции маршрутизации от источника, но при отправке неправильно подсчитывать контрольную сумму IP-заголовка, что приводило к отказу в обработке пакета следующим компьютером.

Было замечено, что контрольные суммы последовательно обрабатываемых пакетов хоть и различаются, но, как правило, отличаются от правильных своих значений на число, равное -4. При достижении IP-пакетом промежуточного узла назначения происходит увеличение указателя следующего адреса в маршруте от источника на 4, что наводит на мысль о том, что разработчики могли забыть учесть это изменение при пересчёте контрольной суммы пакета. Вторым объяснением данной ошибки может быть некорректная делегация подсчёта контрольной суммы IP-заголовка операционной системой сетевой карте.

Вторая попытка данного эксперимента была проведена с заменой маршрутизирующего компьютера на другой компьютер под управлением операционной системы Windows 7 Professional SP1 32-bit v6.1 Build 7600 с другими сетевыми картами. Ошибка повторилась.

Результатом данного эксперимента стало сообщение компании Microsoft о данной проблеме, подробно изложенное на форуме[[4]](#footnote-4) техподдержки разработчиков Windows 7.

### Второй эксперимент

Во втором эксперименте все компьютеры работали по той же схеме под управлением операционной системы Ubuntu 12.10 quantal Linux 3.5.0-17-generic. Настройка маршрутизирующих компьютеров была выполнена в конфигурационном файле /etc/sysctl.conf, были выставлены опции:

* маршрутизации поступающих пакетов

net.ipv4.ip\_forward = 1

* обработки маршрутизации от источника в IP-пакетах

net.ipv4.conf.all.accept\_source\_route = 1

net.ipv4.conf.eth0.accept\_source\_route = 1

net.ipv4.conf.default.accept\_source\_route = 1

С такими настройками операционная система начинала обрабатывать пакеты с маршрутизацией от источника, но делала это некорректно.

При достижении пакетом промежуточного адреса назначения операционная система лишь записывала IP-адрес своего выходного интерфейса на место следующего адреса в маршруте от источника и пересылала пакет дальше. При этом замена адреса назначения на следующий адрес из маршрута и смещение указателя промежуточного узла назначения, требуемые по описанию протокола IP, не производились. Это приводило к тому, что следующий компьютер отказывал в обработке данному пакету и выписывал в системный лог сообщения об отсутствии адреса назначения в маршруте от источника: “ip\_forward(): Argh! Destination lost!”.

Вторая попытка данного эксперимента была проведена с установленной на всех компьютерах операционной системой Ububtu 11.10 oneiric Linux 3.0.0-12-generic и привела к повторению описанной ошибки.

По тексту сообщения в системном логе был найден модуль[[5]](#footnote-5), обрабатывающий опции маршрутизации от источника протокола IP, и написано письмо на рассылку разработчиков ядра Linux[[6]](#footnote-6) с сообщением о несоответствующей описанию протокола IP работе сетевого стека.

## Третий эксперимент

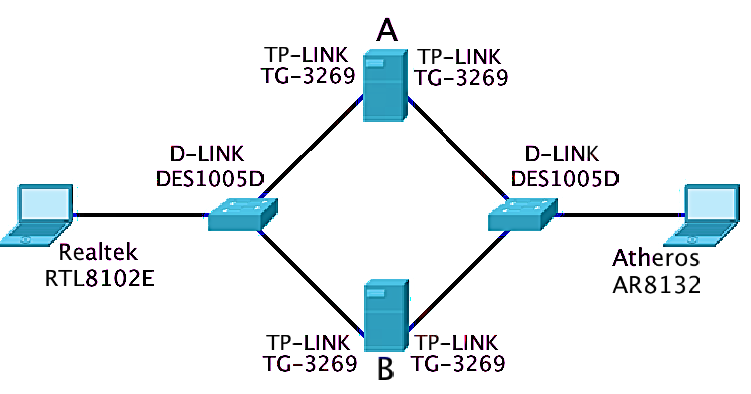
Для третьего эксперимента были построены две локальные сети, имеющие продублированную связь через два осуществляющих маршрутизацию компьютера (A и B). В одну локальную сеть был подключен отправитель данных, во вторую – получатель. Схема соединения сетевого оборудования приведена на рисунке 2.

Рисунок 5 Схема построения сети для третьего эксперимента

Оба маршрутизирующих компьютера имели по две PCI Gigabit Ethernet сетевые карты TP-LINK TG-3269, соединённые с разными локальными сетями, и работали под управлением операционной системы Windows XP Professional SP3 32bit 5.1.2600 в режиме маршрутизации поступающих пакетов. Для этого на них пришлось сделать такие же настройки, что и на маршрутизирующих компьютерах с операционной системой Windows 7 в первом эксперименте.

У отправителя и получателя данных были установлены 100 Fast Ethernet сетевые карты Realtek RTL8102E и Atheros AR8132. Компьютер A был настроен как шлюз по умолчанию у отправителя и получателя.

Эксперимент состоял в передаче одного гигабайта данных по протоколу TCP от отправителя к получателю. Передача выполнялась тремя разными способами:

* **Стандартный**: данные отправлялись без использования опций маршрутизации от источника и передавались через компьютер A.
* **Альтернативный:** данные отправлялись с использованием опций маршрутизации от источника и передавались через компьютер B.
* **Смешанный:** данные отправлялись с использованием опций маршрутизации от источника и передавались одновременно через компьютеры A и B.

Передача с настройками каждого типа выполнялась десять раз. Полученные средние скорости передачи данных приведены в таблице 1.

|  |  |  |
| --- | --- | --- |
| **Способ передачи** | **Среднее время передачи** | **Средняя скорость передачи** |
| стандартный | 169 ± 4 секунды | 6 ± 0.2 МБ/сек |
| альтернативный | 172 ± 4 секунды | 5,9 ± 0.2 МБ/сек |
| смешанный | 120 ± 7 секунд | 8,5 ± 0.5 МБ/сек |

Таблица 1: Средние скорости передачи данных различными способами

Скорость работы Fast Ethernet интерфейсов отправителя и получателя физически имеет ограничение сверху в 100 Мбит/сек. Был проведён тест передачи данных от отправителя к получателю по прямому соединению кабелем. Средняя скорость передачи данных составила 11,3 МБайт/сек, что означает, что максимальное значение пропускной способности сетевых карт отправителя и получателя в ходе эксперимента не было достигнуто. А значит, “узким” местом в сети передачи данных являлись маршрутизирующие компьютеры. С другой стороны, было замечено, что во время передачи данных ни один из компьютеров, участвовавших в передаче, не загружал свои процессоры больше чем на 80%. Такое поведение может быть обусловлено медленной работой драйверов сетевых карт.

Из небольшого различия в пропускной способности стандартного и альтернативного способа передачи данных, можно сделать вывод, что программная обработка сетевого трафика центральным процессором не сильно замедляется обработкой опций маршрутизации от источника. Так как аппаратная обработка сетевого трафика выполняется в разы быстрее программной, а опции маршрутизации от источника из-за сложности своего формата требуют именно процессорной обработки, то скорее всего характер их влияния на пропускную способность канала при аппаратной маршрутизации изменится не в лучшую сторону.

Скорость передачи данных смешанным способом в 1.4 раза больше скорости стандартного способа. Такой сравнительно небольшой прирост пропускной способности соединения связан со слишком медленной работой модуля, конфигурирующего соединения, и говорит о том, что в этом месте разработанную программу требуется усовершенствовать.

# Заключение

## Результаты

В ходе дипломной работы:

* разработан алгоритм определения связности узлов IP-сети в заданном направлении;
* экспериментально определены основные параметры работы алгоритма определения связности узлов IP-сети в заданном направлении;
* разработан алгоритм поиска и выбора альтернативных маршрутов передачи данных в IP-сети в заданном направлении;
* программная реализация разработанных алгоритмов встроена в библиотеку ASIO;
* экспериментально с использованием программной реализации получен прирост пропускной способности соединения в 1.4 раза.

## Развитие работы

Важной частью разработанной программы станет модуль определения состояния используемых маршрутов, их загруженности и работоспособности. Для обработки получаемых от него данных необходимо будет разработать алгоритм динамического определения и настройки пропорций распределения передаваемых данных по каналам.

Также важной частью разработанной программы станет модуль автоматического сбора статистики работы программы. Он позволит программе самой подстраивать заложенные в неё параметры и своевременно реагировать на изменения характеристик сети.

Необходимо будет провести комплексное тестирование разработанной программы в реальных сетях с использованием аппаратных маршрутизаторов и передачей больших объёмов данных для определения её слабых мест и дальнейшей оптимизации.

Конечной целью данной работы станет её внедрение в библиотеку Boost ASIO, используемую для копирования данных между кластерами компании Яндекс.

**Список литературы**

1. RFC 791 Internet Protocol / Information Sciences Institute

// URL: <http://www.rfc-editor.org/rfc/rfc791.txt>

1. RFC 2460 Internet Protocol, Version 6 / S.Deering, Cisco, R. Hinden, Nokia

// URL http://tools.ietf.org/html/rfc2460

1. RFC 1930 Guidelines for creation, selection, and registration of an Autonomous System / J. Hawkinson, BBN Planet, T. Bates, MCI

// http://tools.ietf.org/html/rfc1930

1. Loose Source Routing as a Mechanism fro Traffic Policies / Katerina Argyraki, David R. Cheriton

// URL: <http://www.cs.umd.edu/class/spring2007/cmsc711/papers/lsrr-fdna04-paper.pdf>

1. Internet Control Message Protocol / John Postel

// URL: <http://tools.ietf.org/html/rfc792>

1. The Art of Port Scanning / Gordon Lyon

// URL: <http://nmap.org/nmap_doc.html>

1. RFC 793 Transmission Control Protocol / Information Sciences Institute

// URL http://tools.ietf.org/html/rfc793

1. RFC 768 User Datagram Protocol / John Postel, Information Sciences Institute

// URL: tools.ietf.org/html/rfc768

Приложение 1: Пример графа IP-сети, построенного модулем определения топологии сети разработанной программы



1. *http*://loose-source-bypass.googlecode.com/svn/trunk/ [↑](#footnote-ref-1)
2. http://thyme.apnic.net/current/ [↑](#footnote-ref-2)
3. http://dev.maxmind.com/geoip/legacy/geolite [↑](#footnote-ref-3)
4. <http://social.technet.microsoft.com/Forums/en-US/w7itpronetworking/thread/7f90da44-b8bf-4f18-8014-68204539f051> [↑](#footnote-ref-4)
5. модуль можно увидеть по адресу http://lxr.free-electrons.com/source/net/ipv4/ip\_options.c?v=3.5#L550 [↑](#footnote-ref-5)
6. linux-kernel@vger.kernel.org [↑](#footnote-ref-6)