Федеральное агентство по образованию РФ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет информатики
Кафедра теоретических основ информатики

УДК 681

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, проф., д.т.н. _____
Ю.Л. Костюк «___» _____ 2006 г.

Кравченко Александр Васильевич

# ПОСТРОЕНИЕ И АНАЛИЗ МУЛЬТИСЕРВИСНЫХ СЕТЕЙ ПЕРЕДАЧИ ДАННЫХ, ГОЛОСА И ВИДЕО

Дипломная работа

Научный руководитель,
ведущий инженер ООО «Элекс.Ком» Д. Ю. Белицкий

Исполнитель,
студент группы 1411 А. В. Кравченко

Электронная версия дипломной работы помещена
в электронную библиотеку. Файл
Администратор

Томск-2006

# Реферат

Дипломная работа – 84 с., 38 рис., 6 табл., 27 источников, 2 приложения.

СЕТИ ПЕРЕДАЧИ ДАННЫХ, МУЛЬТИСЕРВИСНЫЕ СЕТИ ПЕРЕДАЧИ ДАННЫХ, AVVID, VOIP, ДИАГНОСТИКА И АНАЛИЗ СЕТЕЙ, ПРОТОКОЛЫ ПЕРЕДАЧИ ПОТОКОВЫХ ГОЛОСА И ВИДЕО, ЗАХВАТ И АНАЛИЗ ТРАФИКА

Объект исследования – мультисервисные сети передачи данных, голоса и видео.

Цель работы – анализ сетевых технологий, применяемых для построения и диагностики мультисервисных сетей передачи данных, голоса и видео.

Методы исследования - теоретический анализ архитектуры, практическое моделирование в лабораторных условиях.

Проанализированы технологии построения мультисервисных сетей передачи данных. Разработано приложение для диагностики и анализа локальных сетей, построена мультисервисная сеть пресс-центра Российско-Германского саммита на высшем уровне, прошедшего в Томске 26-27 апреля.

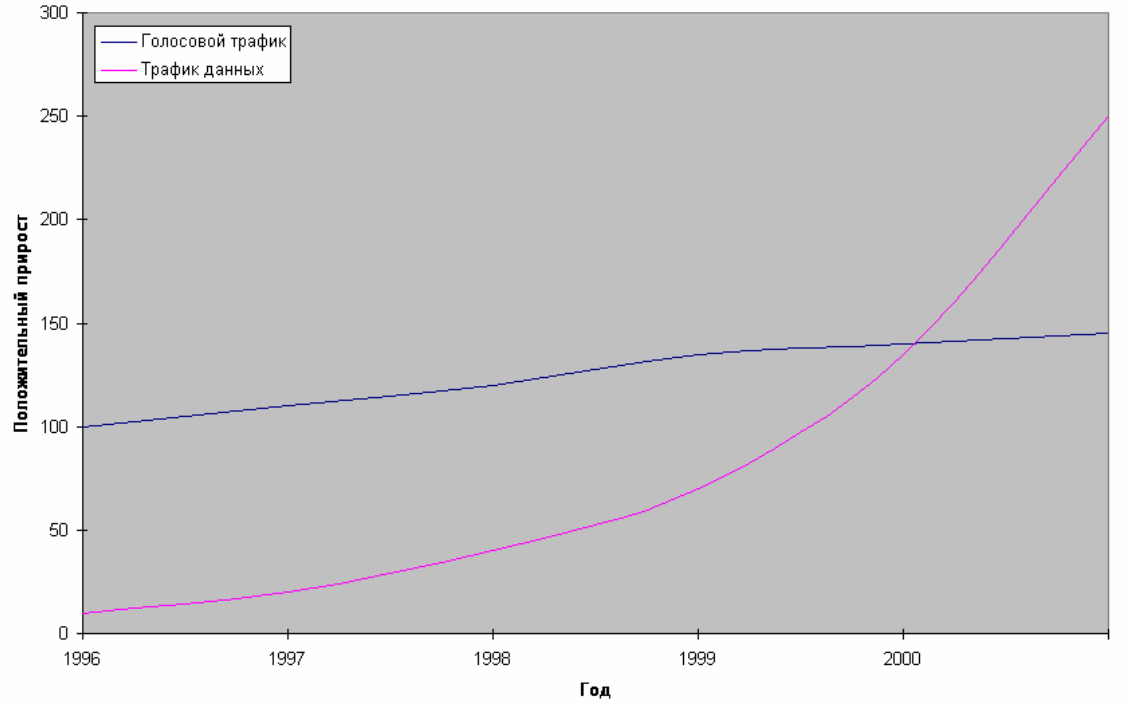
# Содержание

# Введение

Появление мультисервисных сетей способно оказать решающее влияние на развитие индустрии телекоммуникаций и передачи данных. Доставка по единой сетевой инфраструктуре, базирующейся на коммутации пакетов или ячеек, такого разнородного трафика, как данные, голос и видео, является перспективным решением для корпораций и сервис-провайдеров. Исторически корпоративные сети передачи данных, голоса и видео строились независимо, базировались на разных инфраструктурах и технологиях. Обычно в качестве каналов связи использовались выделенные линии, а технологиями передачи данных служили Frame Relay и ATM. Очевидно, что эксплуатация и сопровождение разнородных структур весьма неэффективны, в большей мере нуждам бизнеса соответствуют сети следующего поколения, имеющие возможность передавать все типы пользовательских трафиков на базе сетей с коммутацией пакетов (ячеек), таких, как Frame Relay, ATM или IP.

Современные телекоммуникационные сети используют технологию коммутации каналов и TDM (Time Division Multiplexing) в качестве схемы мультиплексирования. Они проектировались для передачи голосовых потоков и не могут эффективно поддерживать нерегулярный трафик данных. В то же время сегодня темпы роста трафика данных несоизмеримо выше голосового (рис. 1), и операторы столкнулись с проблемой, замены неэффективной TDM-инфраструктуры, с сохранением необходимого качества передачи голоса, решением которой стала постепенная замена TDM-сетей с коммутацией каналов на инфраструктуру пакетных сетей.

Рисунок 1 - Сравнительный график роста трафика данных и голоса



# 1. Архитектура мультисервисных сетей передачи данных

## 1.1. Архитектура Cisco AVVID

Основа построения мультисервисных сетей - архитектура Cisco Architecture for Voice Video and Integrated Data [6, 24, 26]. Это всеобъемлющая архитектура, состоящая из трёх основных блоков (рис. 2):

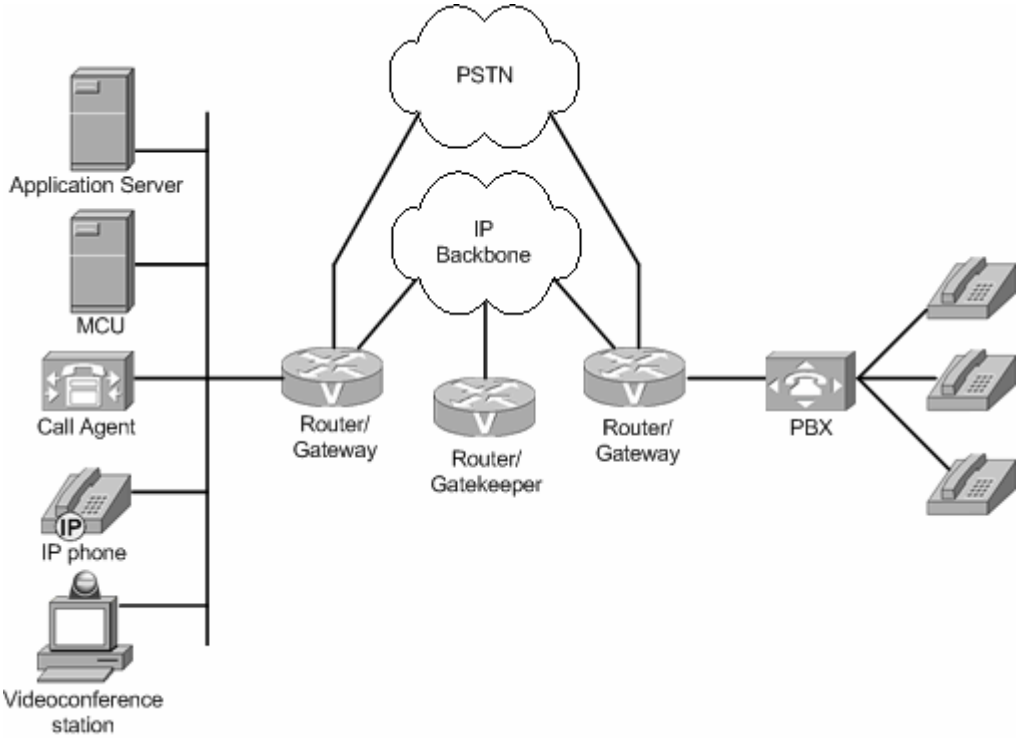
1. Интеллектуальная сетевая инфраструктура на базе протокола IP, включающая в себя маршрутизаторы, коммутаторы, шлюзы и другое сетевое оборудование. IP инфраструктура является основой для дальнейшего внедрения пользовательских приложений и должна обеспечивать поддержку таких жизненно важных для сети сервисов, как безопасность, сетевое управление и механизмов гарантии качества сервиса (QoS, - Quality of Service).
2. Интеллектуальные клиентские места с поддержкой протокола IP, в том числе цифровые IP телефоны, персональные компьютеры со специализированным программным обеспечением для решения различных бизнес-задач, программные эмуляторы телефонов, видео клиенты и так далее.
3. Служебные серверные приложения, в том числе серверы Cisco CallManager, обеспечивающие управление корпоративной системой IP телефонии, корпоративная система директорий, видео серверы и т.д.

Рисунок 2 - Архитектура Cisco AVVID


Мультисервисные сети могут содержать следующие компоненты (рис. 3)

1. IP Phones
2. Gatekeeper
3. Gateway
4. Multipoint control unit (MCU)
5. Call agent
6. Application servers
7. Прочие компоненты, голосовые приложения, системы автоматического ответа (Interactive Voice Response)

Рисунок 3 - Основные компоненты мультисервисной сети

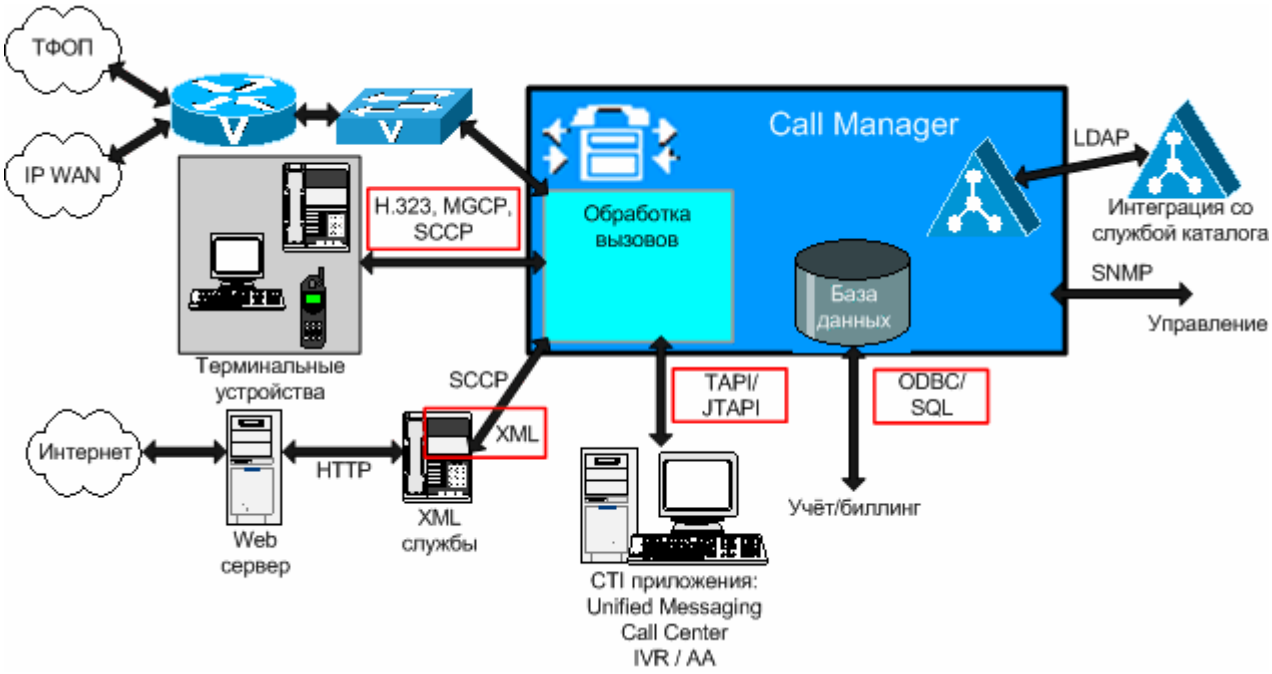



Характерной чертой рассматриваемой архитектуры являются ее распределенная природа, благодаря которой система легко масштабируема. Сеть на базе архитектуры Cisco AVVID может охватывать одно здание или несколько рядом стоящих зданий, объединенных кампусной сетью. Можно обеспечить сервисы телефонии, видео и данных для пользователей удаленных офисов и подразделений, объединенных корпоративной IP сетью.

Другая отличительная особенность архитектуры Cisco AVVID - это ее открытость, - ориентация на использование открытых стандартов (в частности, стандартных протоколов H.323, SIP и MGCP для передачи голоса и видео в сетях IP). Это позволяет обеспечить сопряжение с целым рядом других систем, как традиционной, так и пакетной телефонии, а также с системами передачи данных и видео приложениями, поддерживающими эти стандарты.

Поддержка открытых стандартных протоколов и открытых интерфейсов для разработки приложений (таких как TAPI и JTAPI), обеспечивает возможность написания новых приложений, интегрирующихся в системы на базе Cisco AVVID, а также возможность интеграции приложений, написанных сторонними производителями (рис. 4).

Рисунок 4 - Интеграция с приложениями на основе открытых протоколов и интерфейсов



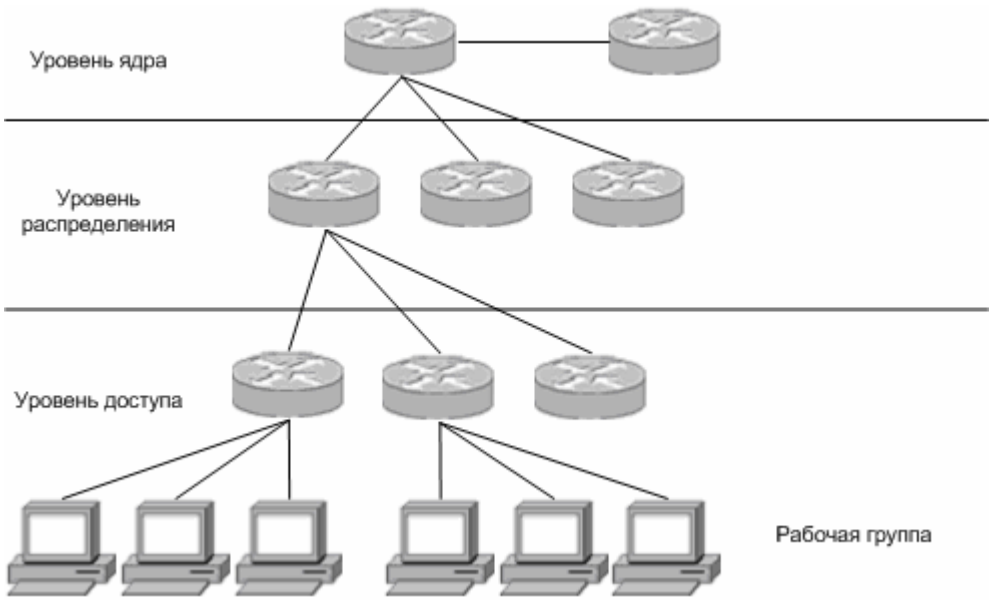

### 1.1.1. Инфраструктура

Как и всякая архитектура, Cisco AVVID имеет устойчивое основание, в виде трёхуровневой модели построения сетей.

**Трёхуровневая модель построения сети.**

Большинство современных сетей построено на основе трёхуровневой модели [24, 26]. Как видно из рис. 5, модель определяет три уровня: уровень ядра, уровень распределения и уровень доступа. Каждый уровень отвечает за реализацию определенных функций. Однако эти уровни являются логическими и не обязательно согласованы с физическими устройствами.

Рисунок 5 - Дизайн сети: трёхуровневая модель



Следование данной модели позволяет значительно упростить построение сети и поиск неисправностей, а также обеспечивает предсказуемость и лучшую управляемость сети. Преимущества трёхуровневого построения, соответствующие требованиям к дизайну сети, либо недостижимы в других моделях, либо требуют значительных усилий для воплощения:

Масштабируемость. Разделение функциональности по слоям позволяет создать естественные точки расширения сети, не оказывая негативного влияния на остальные характеристики.

Лёгкость реализации. Поскольку иерархическая модель разделяет сеть на логическую и физическую составляющие, появляется возможность постепенного построения и ввода в эксплуатацию отдельных участков сети.

Лёгкость поиска неисправности. Как правило, иерархическое построение сети облегчает задачу поиска неисправности, снижая количество возможных циклов.

Предсказуемость. Планирование пропускной способности существенно облегчается в иерархической модели, потребность в пропускной способности возрастают при приближении к ядру.

Управляемость. Предсказуемость потоков данных, масштабируемость, независимость реализации и лёгкость поиска неисправности существенно упрощают управление сетью.

**Уровень ядра.** На самом верху иерархии этот уровень отвечает за быструю и надежную пересылку больших объемов трафика. Единственным предназначением базового уровня является быстрая коммутация трафика.

Если происходит ошибка на уровне ядра, то она влияет на всех пользователей. Следовательно, весьма важно обеспечить высокую надежность на данном уровне. На этом уровне обрабатываются большие объемы трафика, поэтому не менее важно учитывать скорость и задержки.

Из указанных функций уровня ядра, следуют особенности его реализации:

- Ничто не должно замедлять трафик, в том числе списки доступа, маршрутизация между виртуальными локальными сетями VLAN и фильтрация пакетов;
- Не следует реализовывать функции доступа для рабочей группы;
- Следует избегать расширения уровня ядра при росте размеров объединенной сети (например, при добавлении маршрутизаторов). В случае нехватки производительности данного уровня, более предпочтительным выходом является модернизация, а не расширение.

**Уровень распределения.** Уровень распределения иногда называют уровнем рабочих групп. Он расположен между уровнем ядра и уровнем доступа. Основные функции уровня распределения состоят в маршрутизации, фильтрации и доступе к региональным сетям, а также (если необходимо) в определении правил доступа пакетов к уровню ядра. Уровень распределения обязан устанавливать наиболее быстрый способ обработки запросов к службам (например, метод файлового обращения к серверу). После определения на данном уровне наилучшего пути доступа, запрос может быть передан на уровень ядра, где реализован скоростной транспорт запроса к нужной службе. На уровне распределения устанавливается политика сети, а также обеспечиваются возможности гибкого описания сетевых операций. На уровне распределения выполняется несколько функций:

- Реализация инструментов, подобных спискам доступа, фильтрации пакетов или механизму запросов;
- Реализация системы безопасности и сетевых политик, включая трансляцию адресов и установку брандмауэров;
- Перераспределение между протоколами маршрутизации, включая использование статических путей;
- Маршрутизация между сетями VLAN и другие функции поддержки рабочих групп;
- Определение доменов широковещательных и многоадресных рассылок.

**Уровень доступа.** На уровне доступа реализовано управление пользователями и рабочими группами при обращении к ресурсам объединенной сети. Иногда уровень доступа называют уровнем настольных систем. Наибольшая часть необходимых пользователям сетевых ресурсов должна быть доступна локально – для небольших сетей предлагается сохранение отношения трафик локального сегмента/внешний трафик на уровне 80/20, для больших корпоративных сетей существует тенденция к увеличению объёма внешнего трафика – до соотношения 20/80. На уровне распределения выполняется перенаправление трафика к удаленным службам. Для уровня доступа характерны следующие функции:

- Постоянный контроль (из уровня распределения) за доступом и политиками;
- Формирование независимых коллизионных доменов (сегментация);
- Соединение рабочих групп с уровнем распределения.

**Обеспечение требуемого качества обслуживания**

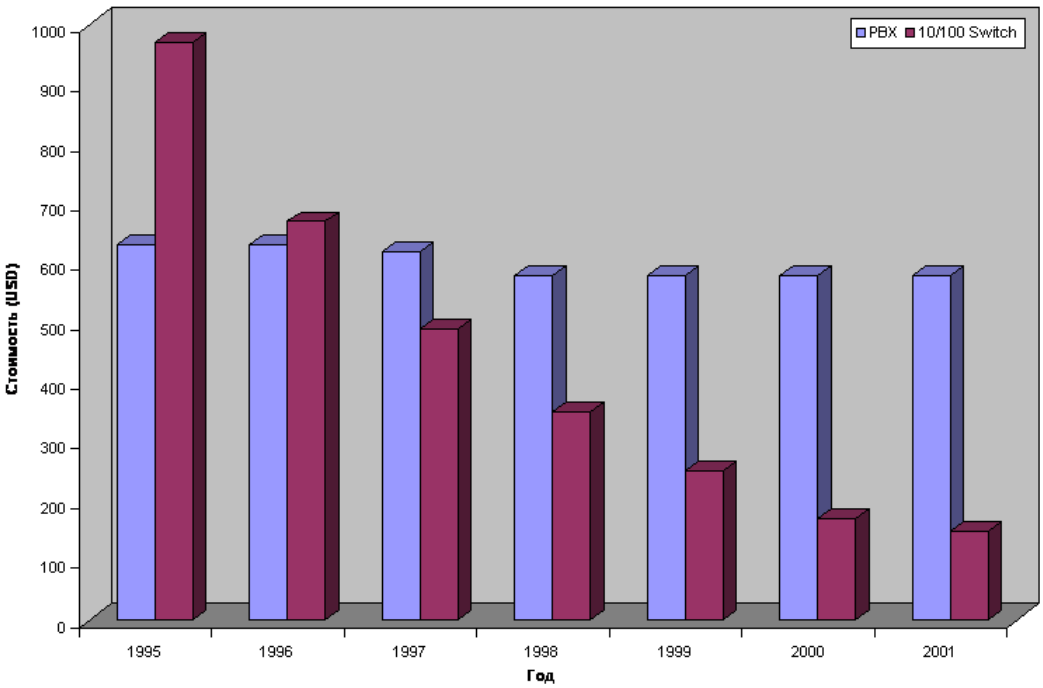
Для перехода от традиционной телефонии к мультисервисным сетям, должны быть обеспечены отказоустойчивость, качество обслуживания (QoS) и пропускная способность, необходимые для поддержки приложений мультисервисной сети, таких как передача потоковых голоса и видео [17, 18].

Например, должны выполняться следующие требования:

- Стандартный кодек G.729 для отсутствия ошибок воспроизведения требует, чтобы потеря пакетов, была значительно меньше 1 процента;
- Спецификация ITU G.114 рекомендует, чтобы задержка при VoIP пакета при пересылке от абонента до абонента не превышала 150 миллисекунд (ms). Для международных звонков приемлемой считается задержка до 300 миллисекунд, особенно при использовании спутниковых каналов. При вычислении этой задержки также учитывается время распространения сигнала вдоль тракта передачи данных;
- Должны быть минимизированы колебания длительности задержки (jitter), для чего используют буферизацию данных. Данное решение увеличивает задержку передачи данных между абонентами и является эффективным только при колебаниях, не превышающих 100 миллисекунд.

Поэтому одним из необходимых условий для внедрения IP телефонии является замен широковещательной среды передачи данных на коммутируемую, однако, эта проблема не актуальна в настоящий момент, что обусловлено существенным падением цен на соответствующее оборудование и практически стопроцентным переходом на коммутируемую среду передачи данных (рис. 6).

Рисунок 6 - Сравнительный график средней стоимости оборудования в расчёте на один порт




Помимо этого для надёжной транспортировки голоса и видео требуется поддержка расширенной приоритизации трафика, многоадресных рассылок, буферизации и компрессии.

Для обеспечения высокого качество передачи голоса требуется, чтобы пакеты VoIP (как сигнального, так и аудио канала) имели приоритет по отношению к другим типам трафика. Необходимо удовлетворение требований по отказоустойчивости, пропускной способности, задержке и колебаниям величины задержки в сети.

**Обеспечение высокой отказоустойчивости**

Традиционная телефония обеспечивает надежное функционирование системы 99.999% времени, это соответствует 5.25 минуты простоя в год. Многие сети передачи данных не обеспечивают такого уровня надёжности. Поэтому одним из основных требований при внедрении VoIP является высокая надёжность и доступности сети.

Меры, направленные на обеспечение отказоустойчивости, могут включать:

- приобретение оборудования и программного обеспечения с высоким показателем MTBF (mean time between failures) – среднее время между сбоями;
- установку дублирующего оборудования;
- прокладку дублирующих линий связи;
- обеспечение бесперебойного электропитания сетевого оборудования, включая оборудование конечных пользователей;
- упреждающее управление сетью и решение проблем до их проявления.

Для полной отказоустойчивости требуется дублирование следующих компонентов:

- серверы и CallManager-ы;
- устройства уровня распределения, такие как маршрутизаторы и многоуровневые коммутаторы;
- устройства уровня ядра, такие как многоуровневые коммутаторы;
- соединения с оператором телефонной связи, WAN, возможно даже через различных провайдеров, голосовые шлюзы;
- источники электропитания и UPS.

**Обеспечение гарантированной пропускной способности**

При переходе к мультисервисной сети требуется обеспечить необходимую пропускную способность для потокового голосового и видео трафика. Это накладывает ограничение на канал передачи данных и сетевое оборудование. Требуемая пропускная способность определяется используемой технологией сжатия аудио или видео данных,

Пропускная способность – это реальный объем полезных данных, переданный от источника до получателя. Предаваемый объём увеличивается за счёт накладных расходов – заголовков протокольных блоков данных различных уровней. Данные также подвержены ошибкам передачи. Объём передаваемых данных ограничен пропускной способность канала, при перегрузке сети возможны потери пакетов, что так же может привести к необходимости повторной передачи.

Для обеспечения требуемой пропускной способности применяются следующие техники:

a. Использование очередей: основывается на передаче пакетов через конкретный интерфейс в соответствии с заданными приоритетами, позволяет обрабатывать интенсивные потоки, управлять нагрузкой сети, приоритизировать трафик, резервировать пропускную способность;
b. Сжатие заголовков: в IP сетях голос передаётся при помощи протокола реального времени Real-Time Transport Protocol (RTP), который переносится протоколом UDP, датаграммы UDP инкапсулируются в пакеты IP. Таким образом, составной заголовок RTP/UDP/IP достигает 40 байт. Это достаточно большая величина, поскольку объем данных, предаваемых в одном пакете, в большинстве случаев составляет 20 байт. Применение сжатия заголовков (CRTP) уменьшает размер заголовка до 2-4 байт.
c. Контроль установления вызова: данный механизм расширяет возможности обеспечения качества обслуживания, обеспечивая защиту голосового трафика от негативного влияния другого голосового трафика путём ограничения количества одновременно установленных вызовов.
d. Фрагментация и чередование: при фрагментации большие пакеты разбиваются на более мелкие, между которыми передаются голосовые пакеты, что позволяет избежать задержек, связанных с выводом больших пакетов в интерфейс.

**Классификация пакетов.** В основе обеспечения качества обслуживания лежит возможность сетевых устройств распознавать и группировать специфические пакеты. Процесс распознавания получил название “классификация пакетов”. После классификации пакет должен быть помечен соответствующим образом, для чего выставляются соответствующие флаги в IP заголовке.

Для распознавания VoIP пакетов сетевые устройства используют адреса источника и получателя в заголовке IP и номера портов UDP источника и получателя в заголовке UDP.

Помимо статической классификации основанной на заголовках протокольных блоков данных 3 и 4 уровней, может быть использован механизм динамической классификации, такой как Resource Reservation Protocol (RSVP).

Классификация пакетов – достаточно ресурсоёмкий процесс, поэтому классификация должна происходить как можно ближе к краю сети. В ядре классификация должна быть максимально упрощена, это достигается за счёт маркирования пакетов - установки байта типа сервиса (Type of Service) в заголовке IP.

Три старших бита байта (ToS) называются битами старшинства IP (IP Precedence). В настоящее время большинство приложений и производителей оборудования поддерживают установку и распознавание битов старшинства IP. Часто для определения дифференцированных классов сервиса (Differentiated Services classes) используются шесть старших битов, называемых Differentiated Services Code Point.

Маркирование пакетов может осуществляется установкой следующих флагов:

- Три бита IP Precedence байта ToS заголовка IP пакета;
- Шесть битов DSCP байта ToS заголовка IP пакета;
- Три бита MPLS Experimental (EXP);
- Три бита Class of Service Ethernet 802.1p;
- Один бит Cell Loss Probability (CLP) ATM.

В большинстве IP сетей, маркирование осуществляется установкой IP Precedence или DSCP, что вполне достаточно для идентификации VoIP трафика.

Классификация и маркирование Voice Dial Peers

Данная техника позволяет классифицировать пакеты VoIP в зависимости от номера, с которым осуществляется соединение.

Классификация и маркирование Committed Access Rate (CAR)

Committed access rate (CAR) – техника использующая лимитирование максимального значения уровня пропускной способности, используемого трафиком. CAR позволяет выставлять различные биты IP Precedence или DSCP в зависимости от того, превышен ли установленный лимит. Однако техника классификации CAR чаще используется для пакетов данных, нежели для пакетов VoIP.

Применение политик маршрутизации (Policy-Based Routing)

Данная техника позволяет маршрутизировать трафик, основываясь на списках доступа (ACL), используемом протоколе, номере порта-источника и так далее. Поскольку данная технология позволяет изменять широкий круг полей пакета или кадра, её применение также возможно для классификации и маркирования пакетов.

Модульный интерфейс QoS командной строки (Mod QoS CLI или MQC)

Данный метод, основанный на применении шаблонов, является наиболее предпочтительным способом классификации и маркирования пакетов. Он позволяет отделить классификацию от политик, обеспечивая возможность конфигурирования различных средств обеспечения качества обслуживания для различных классов трафика. Для классификации трафика применяется class map, а для определения необходимых действий для каждого класса - policy map, которая применяется к входящему или выходящему трафику конкретного интерфейса.

**Организация очередей.** Когда сетевые устройства способны идентифицировать VoIP пакеты, возникает возможность обеспечения требуемого уровня обслуживания (QoS). Для этого могут применяться различные способы организации очередей пакетов на передачу в узлах сети (таблица 1).

Таблица 1. Способы организации очередей пакетов

| Способы организации очередей | Описание | Достоинства | Ограничения |
|---|---|---|---|
| FIFO | Порядок передачи пакетов совпадает с порядком, в котором они были получены. | Простота конфигурирования и высокая скорость работы. | Не обеспечивается приоритетное обслуживание или гарантированная полоса пропускания. |

| Способы организации очередей | Описание | Достоинства | Ограничения |
|---|---|---|---|
| WFQ | Потоки распределяются в различные очереди, где веса используются для определения того, сколько пакетов из этой очереди передаётся подряд.<br>Веса устанавливаются при помощи IP Precedence и DSCP. | Простота конфигурации. По умолчанию используется на соединениях со скоростью до 2 Mbps. | Не обеспечивается приоритетное обслуживание или гарантированная полоса пропускания. |
| Custom Queueing (CQ) | Трафик распределяется в различные очереди, переменной длины. Длина очереди определяется на основе средней длины пакета, максимального размера пакета (MTU) и процента резервируемой полосы пропускания. Таким образом, реализуется статистическое резервирование полосы пропускания. | Возможно приближённое резервирование полосы пропускания для различных очередей. | Не обеспечивается приоритетное обслуживание. Возможно приближённое резервирование полосы пропускания, но ограничено количество очередей. Сложность конфигурирования. |
| Priority Queueing (PQ) | Трафик распределятся по очередям с высоким, средним, нормальным и низким приоритетом. Обслуживание трафика осуществляется в порядке убывания приоритетов. | Обеспечивается обслуживание по приоритетам. | Возможна блокировка низкоприоритетного трафика высокоприоритетным. Не обеспечивается гарантированная полоса пропускания. |
| Class-Based WFQ (CBWFQ) | Для классификации трафика используется MQC. Трафик помещается в очереди с зарезервированной полосой пропускания или очередь “по умолчанию”. Планировщик обслуживает очереди в соответствии с весами, учитывая зарезервированную полосу пропускания. | Способ схож с LLQ за исключением отсутствия приоритетной очереди. Простота конфигурации и возможность резервирования полосы пропускания. | Не обеспечивается приоритетное обслуживание. |
| Priority Queue WFQ (PQ-WFQ) | Приоритет получают UDP пакеты предназначенные для любого из портов внутри задаваемого интервала. | Простота конфигурации. Приоритетное обслуживание пакетов RTP. | Остальной трафик использует WFQ. RTCP трафик не является приоритетным. Не обеспечивается гарантированная полоса пропускания. |

| Способы организации очередей | Описание | Достоинства | Ограничения |
|---|---|---|---|
| LLQ (PQ-CBWFQ) | Для классификации трафика используется MQC. Трафик помещается в очереди с зарезервированной пропускной способностью, приоритетную очередь, или очередь “по умолчанию”. Планировщик обслуживает очереди в соответствии с весами, при этом приоритетный трафик посылается первым и учитывается зарезервированная полоса пропускания. | Простота конфигурации. Возможность обеспечивать приоритетную обработку определённым классам трафика и задавать максимальную используемую полосу пропускания. Есть возможность задавать классы трафика с гарантированной полосой пропускания. | Пока не поддерживаются различные уровни приоритета – весь приоритетный трафик использует одну очередь. Разные классы приоритета могут резервировать различные полосы пропускания. Однако общая приоритетная очередь, разделяемая всеми приложениями, может вызвать колебания длительности задержки. |

Очередь с низкой задержкой (Low Latency Queueing)

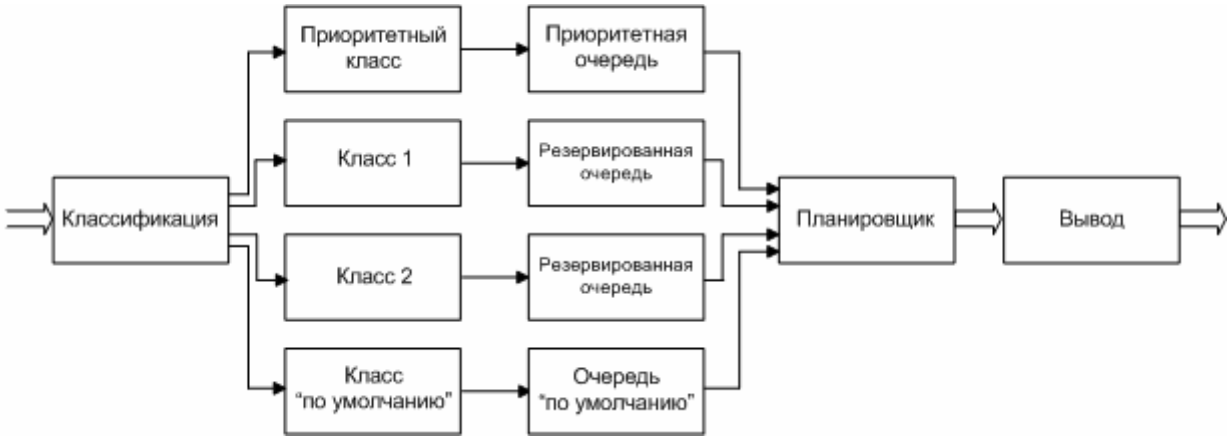
Для VoIP необходима организация приоритетных очередей в узлах сети. Допускается использование любого способа организации очередей, предоставляющего высокий приоритет пакетам VoIP, но ввиду наибольшей гибкости и простоты конфигурирования рекомендуется использование LLQ. LLQ использует метод конфигурирования MQC и позволяет обеспечить приоритет определённому классу трафика, и гарантированную минимальную пропускную способность для других классов. При перегрузке сети приоритетный трафик удерживается в пределах заданного уровня, что позволяет избежать блокировки менее приоритетного трафика.

LLQ позволяет указать длину очереди, при превышении которой маршрутизатор отбрасывает поступающие пакеты. Также имеется класс “по умолчанию”, использующийся для обработки всего неклассифицированного остальными классами трафика. Данный класс может быть сконфигурирован на основе справедливой очереди (fair-queue), это означает, что каждый из неклассифицированных потоков получит примерно равную долю от оставшейся пропускной способности.

Как показано на рис. 7, весь трафик идущий через интерфейс или под-интерфейс (для Frame Relay и ATM) сначала классифицируется с использованием MQC. Существует четыре класса трафика: один приоритетный, два с гарантированной пропускной способностью и один класс “по умолчанию”. Трафик приоритетного класса помещается в приоритетную очередь, классов с гарантированной полосой пропускания в очереди с зарезервированной полосой пропускания. Трафик класса “по умолчанию” может использовать очередь “по умолчанию”, где каждый из неклассифицированных потоков получит примерно равную долю от оставшейся полосы пропускания, или может быть создана очередь с зарезервированной полосой пропускания. Планировщик обслуживает очереди таким образом, что сначала выводится приоритетный трафик, пока не будет достигнута установленная граница используемой данным трафиком пропускной способности, если данная пропускная способность востребована трафиком из зарезервированных очередей (т.е. имеет место перегрузка сети). При переполнении приоритетной очереди в момент перегрузки сети, вновь поступающие приоритетные пакеты будут отбрасываться.

Зарезервированные очереди обслуживаются в соответствии с запрошенной полосой пропускания, которую планировщик использует для вычисления веса. Вес определяет, как часто обслуживается очередь и сколько байт передается за одно обслуживание. Работа планировщика основана на алгоритме weighted fair queueing (WFQ).

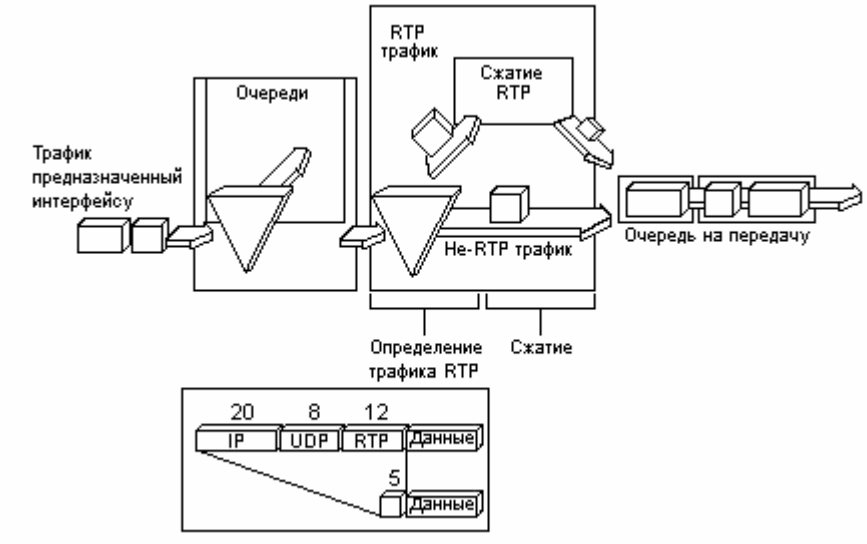
Рисунок 7 - Схема работы LLQ




Сжатие заголовков RTP (cRTP)

CRTP позволяет сжать 40 байтный заголовок IP+UDP+RTP до 2-4 байт, что уменьшает требуемую пропускную способность на соединениях типа точка-точка. Заголовок упаковывается с одной стороны соединения и распаковывается с другой (рис. 8).

Рисунок 8 - Сжатие заголовков RTP




Поскольку cRTP может повлечь сильную загрузку процессора, необходимо ограничивать число потоков, использующих cRTP. cRTP рекомендуется использовать на низкоскоростных соединениях с недостаточной пропускной способностью и небольшим числом VoIP звонков.

**Контроль установления вызова.** Call Admission Control (CAC) применяется к голосовому и видео трафику. В случае если сетевое соединение перегружено пакетами данных, выходом может стать применение очередей, буферизация и отбрасывание пакетов. Трафик задерживается до освобождения интерфейса или отбрасывается, а в последующем пользователь или протокол запрашивают повторную передачу.

Для трафика реального времени, чувствительного к задержке и потере пакетов, такой способ разрешения проблемы приведёт к падению качества обслуживания. В данном случае предпочитают ограничить возможность доступа в сеть, нежели потерять качество.

CAC представляет собой информированный способ принятия решения о достаточности свободных ресурсов для обеспечения требуемого качества передачи голоса. Существует несколько различных механизмов контроля установления вызова:

- локальные – решение об установлении вызова принимается на основе состояния исходящего LAN или WAN интерфейса. Данные механизмы имеют возможность статически ограничить максимальное число одновременно установленных вызовов;
- основанные на измерениях – решение принимается на основе измерения текущего состояния сети, которое проводится посылкой пробных пакетов по заданному IP адресу (обычно это голосовой шлюз адресата). Получатель возвращает пакеты, на основании чего стоится некоторая статистика (обычно задержка и процент потери пробных пакетов), характеризующая состояние сети на данный момент;
- основанные на ресурсах – они делятся на два класса: определяющие количество запрашиваемых и/или свободных ресурсов, и резервирующие ресурсы. Ресурсы представляющие интерес включают пропускную способность соединения, загрузку ЦП, количество памяти.

Локальные механизмы CAC

*Физическое ограничение DS0* – задание количества временных слотов для Time Division Multiplexing интерфейсов. Обладает лёгкостью конфигурирования, но не применим к другим типам интерфейсов.

Достоинства:

- не требует нагрузки ЦП и дополнительной пропускной способности;
- позволяет экономить пропускную способность WAN соединения;
- широко используется.

Недостатки:

- невозможно использовать для IP телефонии в LAN;
- неприменимо для сложной топологии;
- не реагирует на изменения сети.

*Задание максимального числа соединений* – ограничивает максимальное число соединений с каждой группой абонентов (dial-peer). Обладает лёгкостью конфигурирования, но единственным способом ограничить число соединений через данный шлюз является задание определённого числа групп абонентов с ограничением максимального числа соединений для каждой.

Достоинства аналогичны предыдущей технологии.

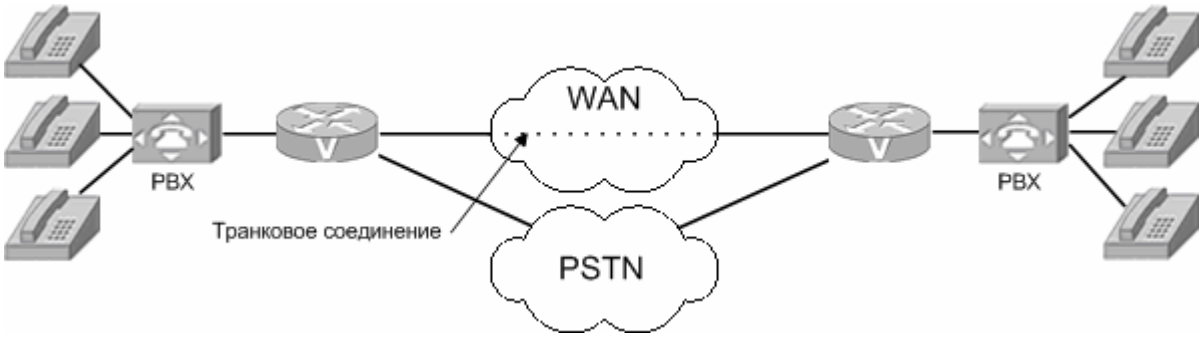
Недостатки:

- можно использовать только при задании групп абонентов (dial-peer);
- неприменимо для сложной топологии;
- не реагирует на изменения сети.

*Задание максимальной используемой пропускной способности* - ограничивает максимальное использование пропускной способности, и используется только для VoFR. При достижении максимального значения всем последующим вызовам будет отказано в соединении.

*Проверка состояния транкового соединения (Trunk Conditioning)* – при выходе из строя транкового WAN соединения, информация об этом передаётся учрежденческой АТС – источнику вызова и вызов может быть направлен по альтернативному пути (рис. 9).

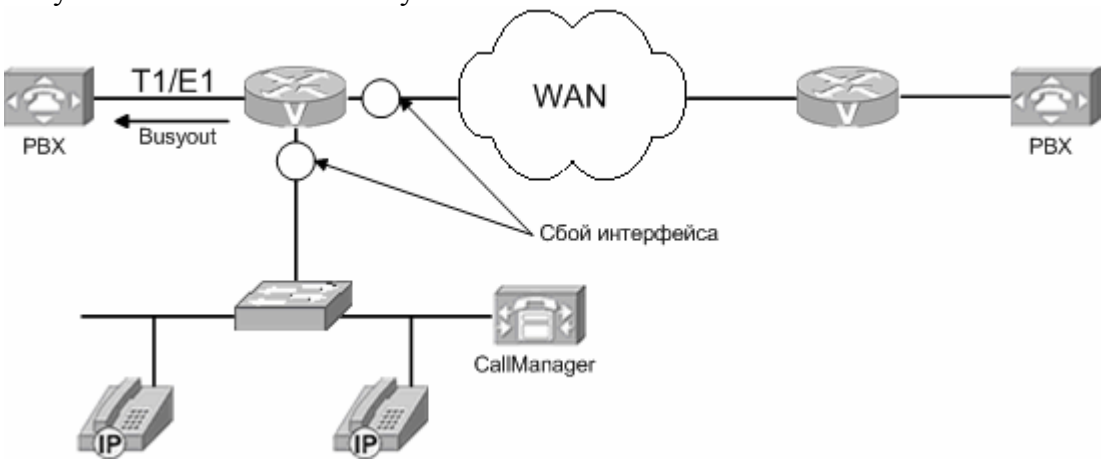
Рисунок 9 - Trunk Conditioning




Особенность механизма является использование постоянного обмена небольшими сообщениями, отслеживающими как состояние WAN соединения между абонентами, так и состояния локальных соединений с Plain Old Telephone System.

*Local Voice Busyout* – аналог механизма Trunk Conditioning, для коммутируемой среды передачи данных. LVBO позволяет объявить транковое соединение PBX со шлюзом как вышедшее из строя, если WAN соединение не обеспечивает приемлемое качество обслуживания. Сигнал Busyout посылается PBX в случае, если происходит сбой любого из отслеживаемых интерфейсов (рис. 10).

Рисунок 10 - Local Voice Busyout




Достоинства:

- не требуется отвергать каждый запрос на соединение индивидуально, что уменьшает postdial задержку;
- не требуется возврата (hairpinning) отвергнутого вызова PBX-инициатору, что достигается использованием нескольких DS0 слотов для одного вызова;
- возможно перенаправление отвергнутых вызовов PBX, которые не поддерживают такой функциональности или не сконфигурированы должным образом.

Недостатки:

- отслеживает Ethernet LAN интерфейсы (не Fast Ethernet);
- используется только для аналоговых или CAS транков.

Механизмы CAC, основанные на измерениях

Данные механизмы опираются на Service Assurance Agent, который обеспечивает измерения задержки и потерь пакетов для принятия решения о сбросе вызова. Хотя явных

измерений пропускной способности вдоль пути пакета не производится, в случае, если имеет место перегрузка сети, следует ожидать высокой задержки и потери пакетов.

SAA – это клиент-серверный протокол, использующий UDP. Клиент конструирует и посылает пробные пакеты. Для точности измерения пакеты SAA строятся также как и пакеты VoIP (IP Precedence и заголовок RTP/UDP/IP), что позволяет учитывать механизмы QoS, существующие в сети. Размер пакетов выбирается соответственно используемому кодеку. Адресат возвращает пакеты отправителю. Пробные пакеты SAA для контроля установления вызова отсылаются случайным образом на порт из верхней части портов UDP, отведенных для передачи аудио (с 16384 до 32767).

Для принятия решения используется показатель Calculated Planning Impairment Factor (ICPIF) - ITU G.113, который представляет задержку и процент потери пакетов в виде одного из чисел, приведённых в таблице 2.

Таблица 2. Интерпретация значений ICPIF (ITU G.113)

| Значение | Оценка качества |
|---|---|
| 5 | Very good |
| 10 | Good |
| 20 | Adequate |
| 30 | Limiting case |
| 45 | Exceptional limiting case |
| 55 | Customers likely to react strongly |

*Advanced Voice Busyout* – расширение LVBO. Как и LVBO обеспечивает подачу сигнала Busyout для PBX, основываясь на локальных данных шлюза, но также поддерживает посылку SAA пробных пакетов по одному или более IP адресам. Возвращаемая информация представляет собой ICPIF или непосредственные величины задержки и процента потери пакетов и служит основанием для сигнализации о занятости сети учрежденческой АТС или отдельному голосовому порту (рис. 11).

Рисунок 11 - Advanced Voice Busyout


Недостатки:

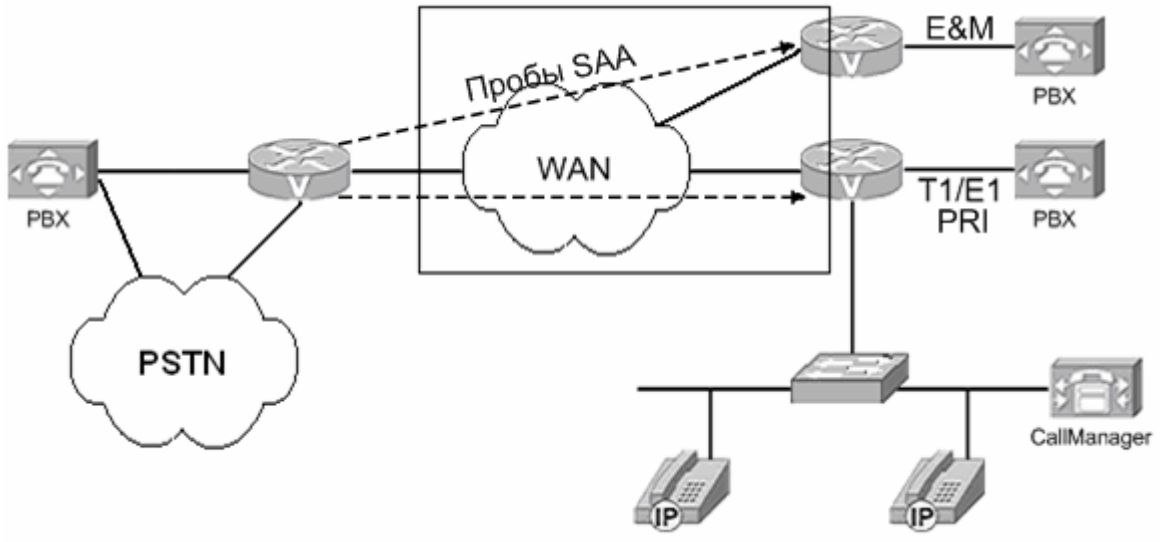
- метод основан на пробах и даёт лишь статистическое, а не абсолютное, решение;
- IP адрес назначения пробных пакетов задан фиксировано и конфигурируется вручную;
- устройство, которому посылаются пробные пакеты должно поддерживать SAA responder;
- мониторинг включает только сети IP, невозможно отслеживать состояние удалённого телефонного транка;

- применение данного метода неэффективно в сетях, для которых характерны большие колебания объёма трафика в короткое время;
- используется только для аналоговых или CAS транков, CCS транки не поддерживаются.

*PSTN Fallback* – в отличие от Advanced Voice Busyout не блокирует транковые соединения и не обеспечивает никакой общей сигнализации PBX о том, что WAN соединение не способно обеспечить требуемое качество обслуживания. Каждое CAC решение принимается по факту поступления запроса на вызов (рис. 12). Данных механизм может принимать решение для любой IP сети, включая Internet. Хотя PSTN fallback нельзя напрямую использовать с IP телефонами и приложениями VoIP для PC, возможно косвенное использование, если данные устройства находятся за маршрутизатором, поддерживающим SAA responder.

Также PSTN fallback не требует статического конфигурирования IP адресов для SAA. Программное обеспечение использует кэш изменяемого размера для хранения последних IP адресов, по которым осуществлялись вызовы. При попадании в кэш CAC решение принимается немедленно, при промахе инициируется серии проб. Таким образом, увеличение postdial delay будет наблюдаться только для первого звонка по каждому заданному IP адресу.

Рисунок 12 - PSTN Fallback




Варианты действий в случае отрицательного решения CAC:

- вызов через другой IP адрес;
- перенаправление вызова через PSTN;
- отказ (reject) вызова PBX/PSTN (BRI/PRI/QSIG);
- возврат (hairpin) вызова PBX/PSTN (аналоговые и CAS протоколы);
- генерация коротких сигналов (reorder tone).

Недостатки:

- механизм применим только для IP сетей;
- при изменении нагрузки на сеть перемаршрутизации установленных соединений не происходит;
- для первого вызова по каждому новому IP адресу возникает увеличение postdial delay;
- метод основан на пробах и даёт лишь статистическое, а не абсолютное, решение;
- применение данного метода неэффективно в сетях, для которых характерны большие колебания объёма трафика в короткое время;

- невозможно измерение пропускной способности;
- возможно использование MD5 аутентификации.

Механизмы CAC, основанные на ресурсах

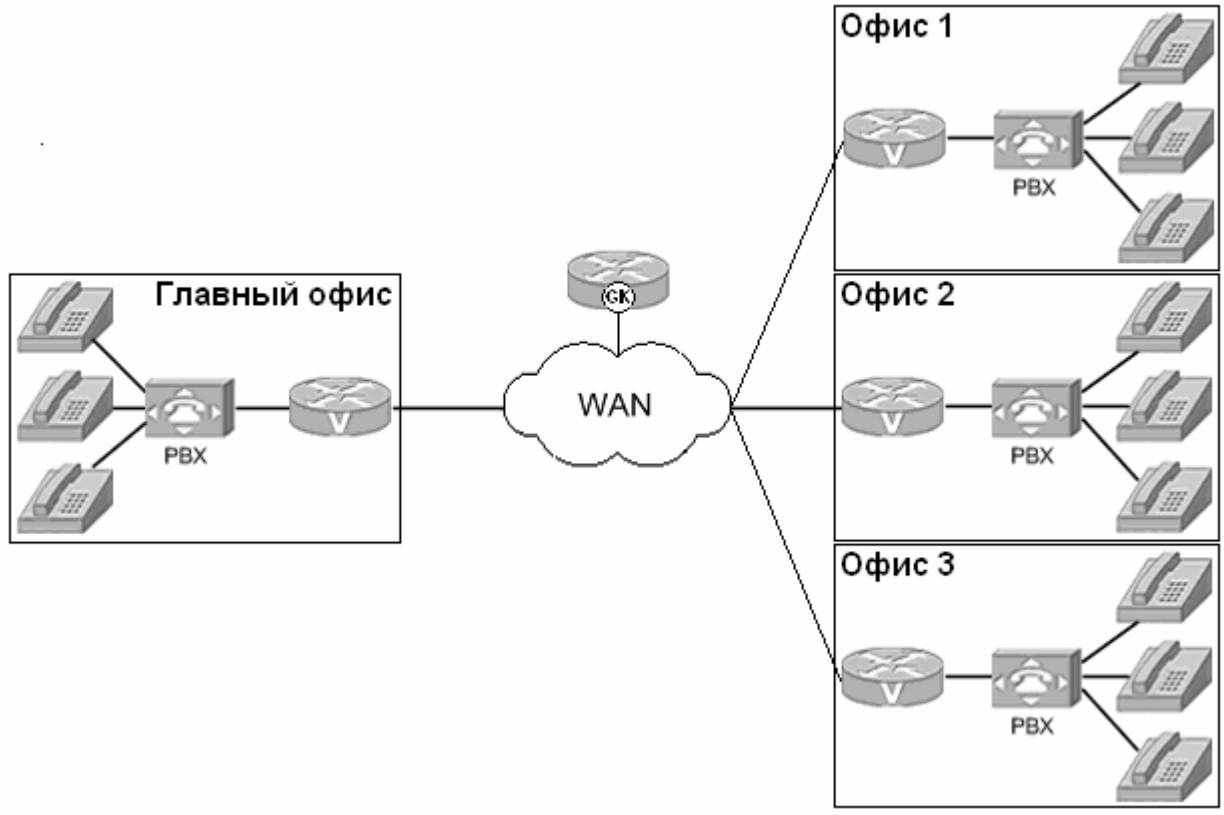
Существует два типа механизмы CAC, основанных на ресурсах:

- механизмы, отслеживающие использование определённых ресурсов и вычисляющих метрику, которая является основой принятия решения CAC;
- механизмы, резервирующие ресурсы необходимые для звонка.

Только механизмы второй категории способны обеспечить QoS на протяжении всего телефонного разговора, прочие механизмы принимают статистическое решение, опираясь на знание о текущем состоянии сети.

*Resource Availability Indication* – представляет собой опциальную функцию протокола H.323v2, обеспечивающую передачу RAS сообщения от конечного (terminating) шлюза gatekeeper-у (рис. 13). Данное сообщение несёт информацию о способности или не способности данного шлюза принять новые звонки, при этом gatekeeper не имеет информации об имеющихся ресурсах шлюза.

Рисунок 13 - Resource Availability Indication



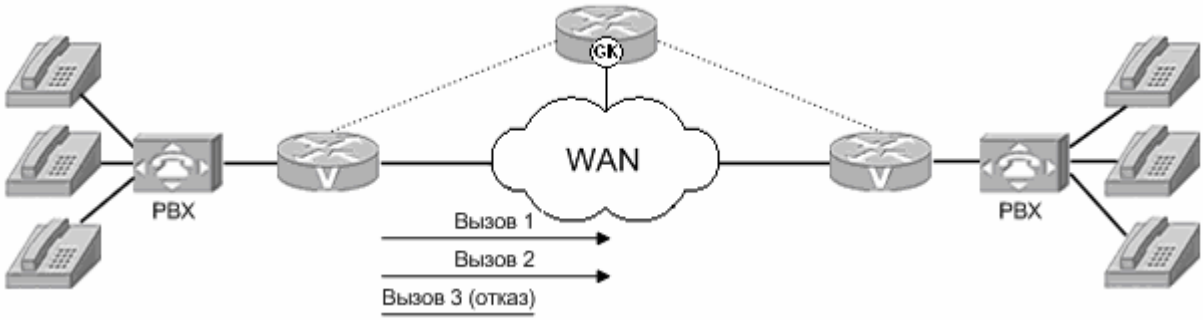

RAI единственный механизм, учитывающий состояние удалённого соединения.

Поскольку RAI обеспечивает передачу сообщения между шлюзом gatekeeper-ом, то данный механизм применим только в H.323 сетях, содержащих gatekeeper.

*Gatekeeper Zone Bandwidth* – этот механизм также специфичен для H.323 gatekeeper сетей, и позволяет устанавливать статические ограничения на используемую пропускную способность внутри определённой зоны, обслуживаемой данным Gatekeeper-ом, и между указанной зоной и любой другой внутри сети. Если запрашиваемый вызов превысит заданное максимальное значение используемой полосы пропускания, происходит отказ в

вызове (рис. 14). При этом gatekeeper не имеет данных о топологии сети или реальном значении используемой различным трафиком пропускной способности.

Рисунок 14 - Gatekeeper Zone Bandwidth




В сетях, где для отказоустойчивости применяется дублирование gatekeeper-ов с использованием Hot Router Standby Protocol, нет разделяемой информационной базы, таким образом, после отказа gatekeeper-а его преемник не будет иметь информации об используемой пропускной способности. И пока данная информация не соответствует действительности, существует возможность установления большего, чем разрешено, количества соединений.

*Resource Reservation Protocol* – единственный механизм CAC, осуществляющий резервацию требуемой полосы пропускания, что позволяет обеспечить не только принятие CAC решения, но и обеспечение QoS на протяжении всего телефонного звонка, независимо от изменяющихся условий функционирования сети.

Резервирование осуществляется в обоих направлениях, поскольку во время разговора информация передаётся в обоих направлениях, решение об установлении вызова принимается шлюзом вызываемого абонента, в зависимости от результатов резервирования.

Отличительными чертами RSVP является:

- возможность обеспечивать качество обслуживания на протяжении звонка;
- знание топологии. RSVP резервирование производится на каждом интерфейсе вдоль пути передачи голосовых пакетов, при этом нет необходимости знать реальную пропускную способность каждого интерфейса. Таким образом, RSVP автоматически учитывает изменения сети;
- для правильного функционирования требуется соответствующая настройка всех сетевых устройств;
- обеспечивая резервирование, данный протокол не учитывает количество уже установленных вызовов.

**Фрагментация и чередование.** Если приоритетный пакет голосового трафика поступает в исходящую очередь в то время, когда передаётся пакет данных из зарезервированной очереди, неизбежна задержка (serialization delay). Учитывая, что пакет данных может иметь размер близкий к MTU (1500 байт для serial и 4470 байт для high-speed serial интерфейсов), величина задержки будет неприемлемой.

Например, для интерфейса со скоростью 64 kbps и MTU 1500 байт, задержка составит (1500 байт * 8 бит/байт) / (64,000 бит/с) = 187.5 мс. При необходимости обеспечить задержку передачи между абонентами не более 150 мс и ограничениями на колебания длительности задержки (jitter). Возникает необходимость уменьшения задержки вывода, что достигается путём рассечения больших пакетов на части, время передачи которых не превышает 10 мс. Размер фрагмента вычисляется (0.01 с * 64,000 бит/с) / (8 бит/байт) = 80 байт. При этом простого фрагментирования недостаточно, поскольку пакет VoIP будет

находится в очереди позади фрагментов большого пакета, требуется переупорядочивание пакетов (рис. 15).

Рисунок 15 - Фрагментация и чередование


Существует три механизма фрагментации и чередования (Link Fragmentation and Interleaving), сравнение которых приведено в таблице 3.

Таблица 3. Механизмы фрагментации и чередования

| Механизмы LFI | Описание | Достоинства | Ограничения |
|---|---|---|---|
| MTU фрагментация с использованием WFQ | Используется для рассечения больших пакетов IP на фрагменты с размером MTU. Применяются WFQ очереди для вставки real-time пакетов между фрагментами. | Простота конфигурирования | Фрагменты собираются в единое целое только приложением-получателем; вследствие чего сеть используется неэффективно. Фрагментация применима только к пакетам IP с неустановленным битом Don't Fragment (DF). Высокая загрузка процессора. |
| Multilink Point-to-Point Protocol (MLP) фрагментация и переупорядочивание | Используется на последовательных соединениях точка-точка, сначала конфигурируется MLP, затем устанавливается максимальный размер фрагмента в мс. | Пакеты фрагментируются с одной стороны соединения и собираются с другой. Несколько каналов могут быть объединены в один виртуальный канал. | Доступно только на каналах PPP. Поддерживаются решения PPP over Frame Relay и PPP over ATM. |
| Фрагментация Frame Relay (FRF.12) | Применяется на каналах Frame Relay PVC, размер фрагмента задаётся при помощи map class. | Пакеты фрагментируются с одной стороны канала PVC и собираются с другой. | Возможно только на каналах Frame Relay PVC с доступной командой конфигурирования интерфейса frame-relay traffic-shaping. |

### 1.1.2. Приложения

Все приложения можно разделить на две категории:

- служебные серверные приложения, в том числе серверы Cisco CallManager, обеспечивающие управление корпоративной системой IP телефонии, корпоративная система директорий, видео серверы и т.д.;
- Современные пользовательские приложения, возникшие благодаря развитию интегрированных систем с поддержкой голоса, видео и данных, - например, система унифицированной обработки сообщений (Unified Messaging [23]) или интеллектуальные центры обработки вызовов. Внедрение подобных приложений позволяет обеспечить дополнительные возможности для пользователей/абонентов корпоративной телекоммуникационной сети, повысить удобство и эффективность использования системы.

### 1.1.3. VOIP оборудование

**IP телефоны.** Обеспечивают непосредственное взаимодействие с пользователем.

Для функционирования, IP телефоны требуют подключения к сети передачи данных и к источнику питания.

Подключение к сети передачи данных производится кабелем Cat 5 (и выше), имеющим разъём RJ-45.

Возможны следующие варианты подключения:

1. Один кабель (рис. 16). Большинство организаций использует именно этот вид подключения. Причиной этого служит простота установки, отсутствие дополнительного расхода портов коммутатора и необходимости изменение кабельной инфраструктуры.
2. Раздельное подключение (рис. 17). Данный тип подключения обеспечивает физическое разделение между сетью передачи голоса и данных.
3. Подключение к разным коммутаторам (рис. 18). Этот вид подключения позволяет избежать расходов на замену существующих коммутаторов и обеспечивает полное разделение сетей передачи голоса и данных. При этом появляется возможность обеспечить электропитание IP телефонов по кабелю передачи данных без замены существующего оборудования, и уменьшается количество коммутаторов, нуждающихся в бесперебойном электропитании.

Рисунок 16 - Подключение одним кабелем

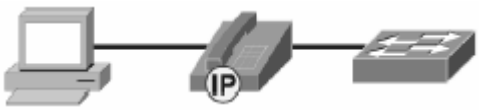



Рисунок 17 - Раздельное подключение

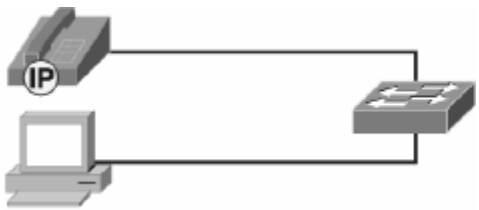
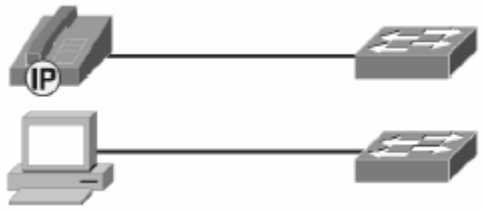


Рисунок 18 - Подключение к разным коммутаторам




Также требуется обеспечить электропитание IP телефонов. Существует два возможных способа:

1. Использование внешнего блока питания отдельно для каждого телефона;
2. Технология Inline Power. Данная технология обеспечивает возможность подачи электропитания для телефонных аппаратов по стандартной витой паре пятой (и выше) категории от Ethernet-коммутатора. Связанная с ней технология автоопределения подключения IP телефона позволяет предотвратить возможность повреждения других типов Ethernet-устройств при их подключении к коммутатору с поддержкой Inline Power: прежде чем включить подачу питания, коммутатор проверяет, является ли подключенное устройство IP телефоном.

Технология подачи питания для IP телефонов Inline Power обладает двумя важными преимуществами: во-первых, не потребуется локальная розетка электропитания для каждого телефонного аппарата, и, во-вторых, этот способ также позволяет централизовать средства управления и обеспечения надёжности электропитания.

Поскольку требуется обеспечение надёжной работы телефонной сети при сбоях электропитания (до 4 часов работы в автономном режиме), необходимо использование источников бесперебойного питания (UPS); при этом UPS могут использоваться только для коммутаторов, поддерживающих технологию Inline Power и других важных сетевых устройств и серверов (в том числе шлюзов и серверов Call-Manager).

Существует два способа реализации технологии Inline Power:

1. Использование модулей/устройств, поддерживающие технологию Inline Power.
2. Устройство Catalyst Inline Power Patch Panel, представляющее собой 48-портовый кросс с поддержкой технологии Inline Power. Может использоваться для подачи питания IP телефонам по сети Ethernet в случае, если установленные коммутаторы локальной сети не поддерживают соответствующую технологию.

Коммутаторы Cisco, поддерживающие технологию подачи питания по стандартной витой паре категории 5, используют для подачи энергии от коммутатора пары 2 и 3. Этот метод обеспечения питания иногда называют фантомным, поскольку силовые сигналы проходят по тем же двум парам, что используются для передачи сигналов Ethernet.

Устройство Catalyst Inline Power Patch Panel, в отличие от коммутаторов, использует для подачи питания пары 1 и 4, не используемые в сети Ethernet. В последнем случае кабельная система, соединяющая кросс с рабочими местами, на которых будут установлены IP телефоны, должна содержать все 4 пары UTP Cat5.

**Gatekeeper.** Обеспечивает контроль установления вызова, трансляцию адресов, контроль пропускной способности и управление.

**Gateway.** Обеспечивает взаимодействие между VoIP сетями и сетями других типов, в частности, общественной телефонной сетью. Также обеспечивается физическое подключение для локальных аналоговых телефонов, факсов и частных телефонных систем (PBX).

**Multipoint control unit (MCU).** Обеспечивает передачу real-time данных для множества географически разделённых участников конференции. Возможно переключение или смешивание медиа-потоков.

### 1.2. Модели построения систем IP телефонии

Встречается три основных модели построения сетей Cisco IP телефонии.

Простейший вариант представляет из себя локальную/кампусную сеть с интеграцией голоса и данных (а также, возможно, видео приложений).

В этом случае сетевая инфраструктура представлена коммутируемой сетью на базе технологий Ethernet / Fast Ethernet / Gigabit Ethernet. Пользовательские IP телефоны подключаются в пределах локальной/кампусной сети и работают под управлением сервера Cisco CallManager. Один сервер Cisco CallManager может поддерживать до 2500 IP телефонов. В целях масштабирования системы и для обеспечения отказоустойчивости серверы Cisco CallManager могут быть объединены в кластер.

В локальной/кампусной сети экономия полосы пропускания не является критичной, поэтому для голосовых звонков в пределах локальной сети сжатие голоса обычно не используется.

Серверы пользовательских приложений, таких как система голосовой почты или интерактивных голосовых меню, расположенные в пределах кампусной сети, обеспечивают дополнительные сервисы для абонентов системы.

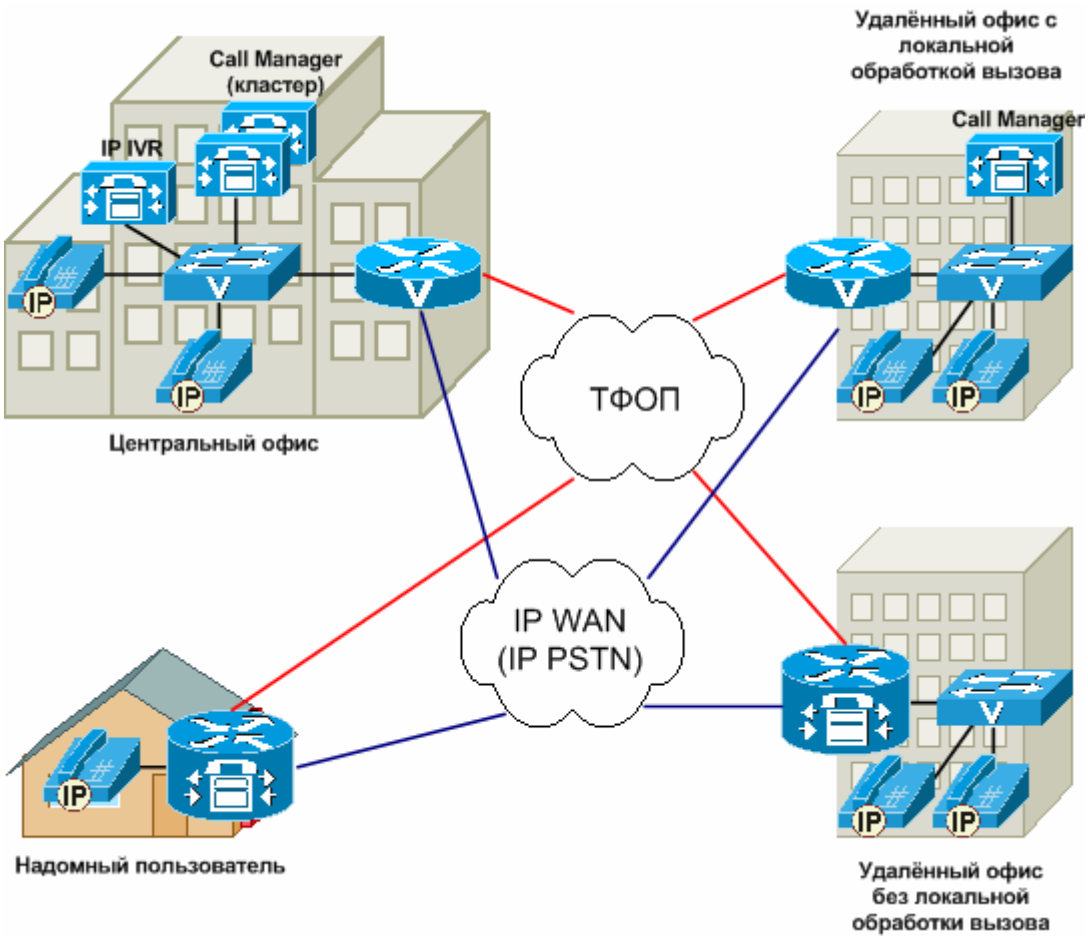
Основные характеристики модели построения сети IP телефонии для одного здания или кампуса (нескольких зданий, объединенных высокоскоростной локальной сетью):

- для организации системы IP телефонии используется сервер Cisco CallManager или кластер серверов Cisco CallManager (для обеспечения масштабируемости и отказоустойчивости решения в пределах кампусной сети);
- на одном сервере Call Manager поддерживается до 2500 телефонов;
- поддерживается до 10000 IP телефонов на кластер Cisco CallManager;
- для дальнейшего масштабирования сети возможность использование нескольких кластеров Cisco CallManager;
- максимальное количество серверов Cisco CallManager в кластере – восемь (4 сервера для основной обработки вызовов, два для резервной обработки, один сервер базы данных и один TFTP сервер);
- для подключения к телефонной сети общего пользования (ТФОП), подключения аналоговых телефонов и факсовых аппаратов и стыковки с существующими учрежденческими АТС используются голосовые шлюзы;
- ресурсы голосовых сервисных модулей используются для организации аудиоконференций;
- для всех голосовых звонков используется кодек G.711 (несжатый голос);
- для обесепечение качественной работы различных приложений используются коммутаторы, поддерживающие необходимые средства обеспечения качества сервиса (QoS).

Один из наиболее распространенных вариантов построения системы IP телефонии представляет собой распределенную систему, обеспечивающую сервисы корпоративной IP телефонии не только для центрального офиса, но и для удаленных подразделений/офисов, подключенных к корпоративной IP сети с обеспечением необходимых механизмов качества сервиса (QoS).

В такой схеме сервер CallManager, расположенный в центральном отделении, управляет установлением телефонных соединений и функционированием телефонных аппаратов, расположенных в удаленных точках в пределах корпоративной IP сети (рис. 19).

Рисунок 19 - Распределённая схема построения систем IP телефонии




Подобная архитектура имеет ряд достоинств, среди них:

- простота и экономичность внедрения телефонии для небольших удаленных отделений;
- возможность централизованной настройки и управления телефонной системой;
- простота организации доступа удалённых абонентов к современным сервисам телефонии, развернутым в центральном отделении, таким как сервисы голосовой почты/унифицированной обработки сообщений, доступ к автоматическим телефонным справочникам с IP телефона и т.д.;
- возможность использования ресурсов корпоративной сети передачи данных для установления телефонных соединений между различными отделениями, объединенными сетью IP телефонии. При этом возможна экономия на оплате междугородних телефонных разговоров между различными отделениями и повышение эффективности использования каналов WAN за счет использования единого набора каналов для передачи трафика голоса и данных;
- нет необходимости иметь опытный персонал службы технической поддержки в каждом удаленном подразделении/офисе.

При использовании подобной схемы построения сети должна быть обеспечена возможность локальной обработки вызовов в удаленном отделении на случай потери связи между удаленным и центральным отделением, например в случае сбоя канала WAN. Для этой цели можно использовать средства отказоустойчивой телефонии для удаленных офисов (Survivable Remote Site Telephony).

Основные характеристики распределенной модели построения сети IP телефонии с централизованной обработкой вызовов:

- сервер Cisco CallManager или кластер серверов Cisco CallManager, расположенный в центральной точке сети используется для управления локальными телефонами и телефонами, находящимися в удаленных офисах;
- на одном сервере Call Manager поддерживается до 2500 телефонов;
- удаленные офисы подключаются к корпоративной IP сети с обеспечением необходимых механизмов качества сервиса (QoS);
- для подключения к телефонной сети общего пользования (ТФОП), подключения аналоговых телефонов и факсовых аппаратов и стыковки с существующими УАТС используются голосовые шлюзы;
- голосовые шлюзы могут располагаться как в центральной, так и в удаленных точках сети IP телефонии;
- для организации конференций и транскодинга (перекодирования голоса из низкоскоростного кодека в высокоскоростной) можно использовать голосовые сервисные модули (расположенные в сети центрального офиса);
- в пределах локальной сети возможно использование кодека G.711 (несжатый голос);
- для экономного использования полосы пропускания на каналах WAN может быть использовано сжатие голоса (кодек G.729);
- Cisco CallManager контролирует использование полосы пропускания на каналах WAN между удаленными офисами и принимает решение о разрешении/запрете установления телефонного соединения на основе информации о наличии свободной полосы пропускания (call admission control);
- поддержка механизмов обеспечения качества сервиса (QoS) в пределах распределенной IP сети является критично важной для обеспечения качественной работы различных приложений (это особенно важно для голосовых приложений).

Третий вариант построения сетей IP телефонии предусматривает использование собственных управляющих серверов Cisco CallManager и серверов приложений в каждом офисе. Такая модель применяется для сетей, объединяющих крупные и средние офисы или в случае, когда имеются специфические требования к сервисам телефонии для конкретных офисов, их надежности и быстродействию.

В таком варианте построения сети для организации взаимодействия между серверами/кластерами серверов CallManager, расположенными в центральном и удаленных офисах компании, может использоваться H.323 gatekeeper. Gatekeeper может также использоваться в этой модели для целей контроля за установлением телефонных соединений (call admission control).

Один H.323 gatekeeper может обеспечить взаимодействие до 100 кластеров Cisco CallManager. Возможна также иерархическая модель с построения сети с использованием Directory gatekeeper'a. Это обеспечивает возможность масштабирования системы до многих сотен тысяч абонентов.

Возможно также использование смешанных моделей построения сети IP телефонии, включающих существующие учрежденческие АТС.

### 1.3. Протоколы IP телефонии

Короткая, но богатая событиями история развития IP-телефонии привела к тому, что сегодня в реальных сетях VoIP сосуществуют и конкурируют между собой несколько семейств сигнальных протоколов, которые регламентируют управление мультимедиа-вызовами и передачу медиа-трафика в IP-сетях.

Помимо протокола сигнализации другой важной составляющей частью IP телефонии является протокол RTP (Real Time Protocol), который обеспечивает сквозной сетевой транспорт для приложений требующих передачи потоковых данных в реальном времени, таких как аудио и видео.

RTP является критическим компонентом VoIP, обеспечивая для получателя возможность переупорядочивания и хронометража пакетов перед воспроизведением. Заголовок RTP содержит временные отметки и последовательные номера, позволяющие буферизовать пакеты и устранять колебания длительности задержки.

Протокол RTCP отслеживает качество передачи данных и обеспечивает управляющую информацию. Для устройств вовлечённых в RTP сессию RTCP обеспечивает механизм обмена управляющей информацией и информацией о состоянии сессии. RTCP отслеживает такие показатели качества как: количество переданных и потерянных пакетов, задержка и колебание длительности задержки.

### 1.3.1. Сигнальные протоколы

На сегодняшний момент в реальных сетях VoIP представлены три основных семейства сигнальных протоколов - H.323, SIP и MGCP. Протоколы всех трех перечисленных семейств регламентируют управление мультимедиа-вызовами и передачу медиа-трафика в IP-сетях, но при этом реализуют три различных подхода к построению систем телефонной сигнализации.

#### Набор рекомендаций H.323

Исторически первый и самый распространенный в настоящее время - это введенный Международным союзом электросвязи (МСЭ) набор рекомендаций H.323. H.323 стал плодом деятельности разработчиков протоколов мультимедийной связи в сетях ISDN (H.320). Первая версия этого протокола была принята МСЭ в 1996 году и, по сути, была попыткой перенести телефонную сигнализацию ISDN Q.931 на IP-соединения, "наложить" традиционную телефонию на сети передачи данных. Рекомендации H.323 достаточно подробно описывают способы организации мультимедийных конференций, охватывая сервисы передачи голоса, видео и компьютерных данных в пакетных сетях с негарантированной доставкой. К настоящему времени принята уже четвертая версия этого набора рекомендаций. К основным компонентам набора относятся описанные ниже протоколы.

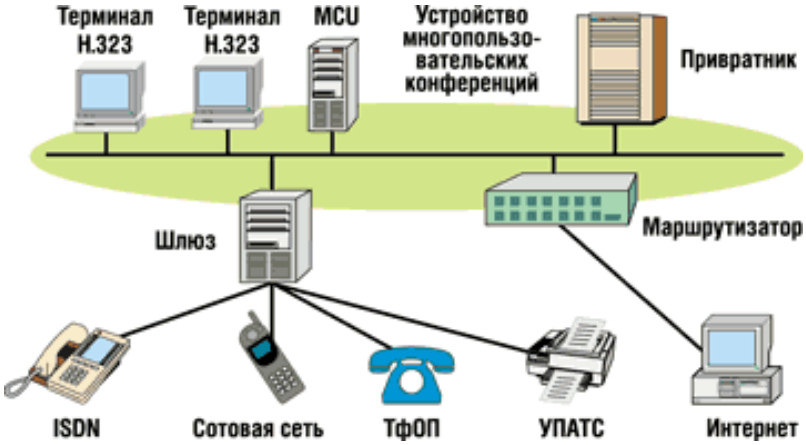
H.225 - полный аналог протокола Q.931 в сетях ISDN; описывает процесс установления, поддержки и завершения соединения. Обмен сообщениями происходит по протоколу TCP.

RAS (Registration, Admission, Status) - отвечает за регистрацию устройств в сети, контроль доступа к ресурсам, контроль полосы пропускания, необходимой для сеанса связи, и контроль состояния устройств в сети. Работает по протоколу UDP. H.245 - отвечает за обмен информацией, необходимой для согласования параметров логических каналов для передачи медиа-потоков, то есть собственно голоса или видео. Сюда входит, к примеру, согласование кодеков, номеров UDP-портов и так далее. Обмен происходит по протоколу TCP.

H.450.x (появившийся в четвертой версии H.323) - отвечает за обеспечение таких дополнительных или интеллектуальных функций, как Hold, Transfer и так далее.

Архитектура H.323 (рис. 20) весьма проста и состоит всего из четырех функциональных компонентов, ни один из которых не является обязательным.

Рисунок 20 - Архитектура H.323




Терминал (H.323 Terminal) - абонентское устройство, способное обеспечивать связь (голосовую, видео и т. д.) с другими терминалами, шлюзами или устройствами многопользовательских конференций.

Шлюз (H.323 Gateway) - центральное понятие IP-телефонии. Данное устройство обеспечивает взаимное сопряжение телефонной сети с IP-сетью. При этом предоставляется поддержка разных протоколов и интерфейсов сетей обоих типов. Если выход в телефонную сеть не требуется, то данный компонент не нужен, а терминалы могут связываться друг с другом напрямую.

Привратник (H.323 Gatekeeper, GK) - управляющий элемент H.323 сети, обеспечивающий ее масштабируемость, централизацию управления и настроек, а также трансляцию телефонных префиксов и идентификаторов (H.323 ID) в IP-адреса шлюзов или H.323 терминалов. Кроме того, привратник отвечает за управление доступом (Admission Control) при регистрации шлюзов и терминалов, контроль установления вызова (Call Admission Control), управление полосой пропускания и маршрутизацию вызовов. Привратник управляет подчиненной ему частью сети (зоной) через RAS - протокол взаимодействия со шлюзами. Предусмотрено объединение привратников в группы, управлять которыми можно с помощью выделенного привратника - Directory Gatekeeper.

Устройство многопользовательских конференций (H.323 Multipoint Conference Unit, MCU) - управляет проведением многопользовательских конференций, согласует параметры соединения всех участников в режиме централизованной, децентрализованной или комбинированной конференции. Возможно переключение или смешивание медиа-потоков.

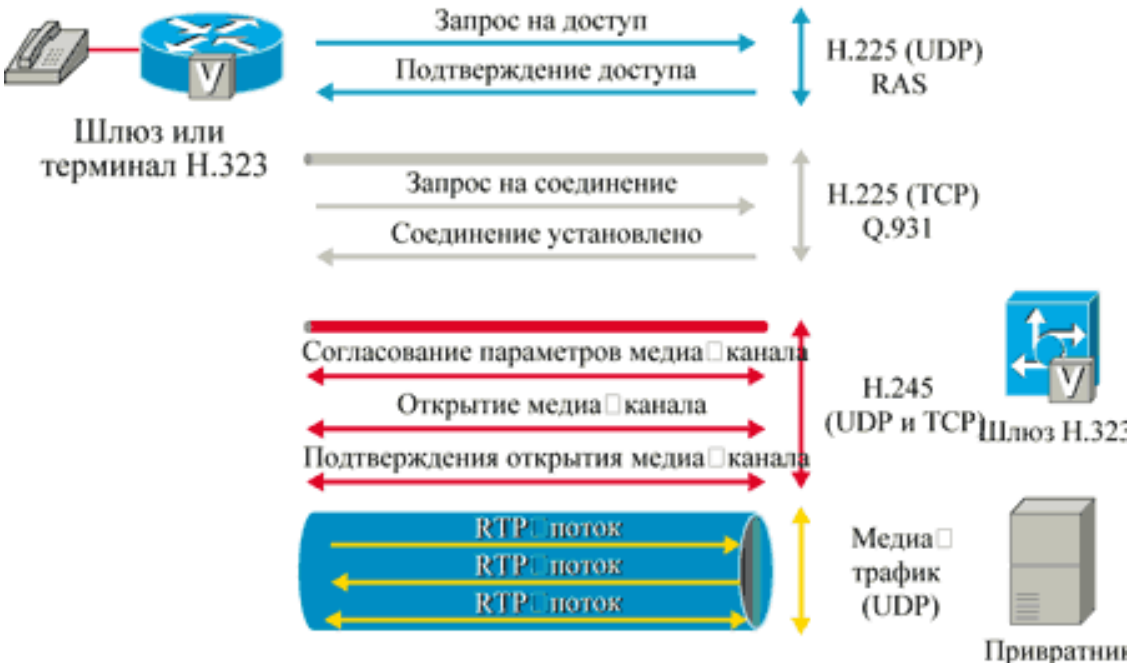
Обмен сообщениями между компонентами сети H.323 происходит в двоичном формате (ASN.1), для анализа которого нужен транслятор из двоичного формата в текстовый (ASN parser). В рекомендациях H.323 определено несколько различных способов адресации:

- телефонные номера в формате E.164, т. е. только символы из набора "0123456789#*,";
- H.323-идентификатор (H323-ID) - произвольный набор символов Unicode;
- универсальный идентификатор ресурса в формате URL (URL-ID);
- IP-адрес с номером порта, например, 10.2.3.4:1720;
- адрес электронной почты (Email-ID).

В наиболее общей форме сценарий соединения по протоколу H.323 выглядит как ряд последовательных шагов (рис. 21). Вначале для установления соединения терминал обнаруживает привратника и регистрируется у него по протоколу RAS. Затем происходит установление сигнального канала по протоколам RAS и H.225. На следующем этапе выполняется согласование параметров оборудования, обмен информацией о его функциональных возможностях и открытие логических каналов по протоколу H.245. Только

после этого происходит передача медиа-трафика по протоколам RTP/RTCP, а по ее окончании - завершение соединения.

Рисунок 21 - Сценарий соединения по протоколу H.323.




**Протокол SIP**

Следующий по распространенности протокол IP-телефонии - SIP (Session Initiation Protocol); он описан в рекомендациях RFC 2543. SIP регламентирует установление и завершение мультимедийных сессий - сеансов связи, в ходе которых пользователи могут говорить друг с другом, обмениваться видеоматериалами и текстом, совместно работать над приложениями и так далее. SIP и сопутствующие ему протоколы родились и развиваются в рамках IETF - главного органа стандартизации Интернета. Первая версия протокола SIP была принята в марте 1999 года, на три года позже, чем H.323, но благодаря интенсивному развитию этого направления сегодня набор рекомендаций RFC (базовых официальных документов IETF), имеющих отношение к SIP-архитектуре, насчитывает десятки, если не сотни документов.

SIP очень похож на протокол HTTP, поскольку разрабатывался по образу и подобию широко известных спецификаций HTTP и SMTP. По сути, это клиент-серверный протокол, работа которого состоит из череды запросов и ответов, причем все SIP-заголовки передаются в формате ASCII-текста, а потому легко читаются. SIP позволяет использовать логическую адресацию (URL) на базе протокола TCP или UDP. Проще всего в качестве адреса в сети SIP задавать адреса электронной почты. При этом допускается применение разнообразных параметров, определяющих функциональность SIP-адреса или тип протокола связи. Например, можно указать, что соединение осуществляется с обычным телефонным номером сети общего пользования - sip:tel:+70957852525, и дополнить его добавочным номером postd=pp521, или определить параметры модемной связи - modem: +70957852526;type=v32b?7e1;type=v110.

SIP имеет несколько комплементарных протоколов, которые служат для реализации дополнительных возможностей. Наиболее важный из них - SDP (Session Description Protocol, RFC 2327), протокол согласования таких параметров сеанса связи, как виды кодеков, номера UDP-портов и так далее. SDP обеспечивает изменение параметров сеанса связи

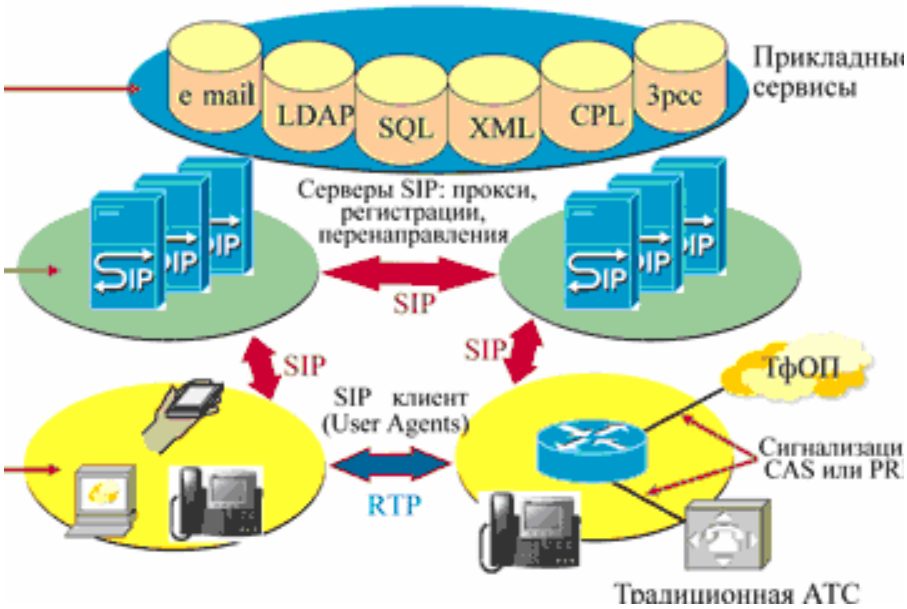
непосредственно во время сеанса. Перенос сообщений SDP основан на протоколе Session Announcement Protocol (SAP, RFC 2974).

Другой пример комплементарного протокола - SIMPLE (SIP for Instant Messaging and Presence Levering Extension). Фактически это расширение SIP, служащее для предоставления информации о событиях (presence) и для рассылки "мгновенных" сообщений (instant messaging).

Следует также упомянуть SIP-T (Trunk) - протокол переноса сообщений SS7 в виде MIME-объектов между контроллерами сигнализации, а также SIGTRAN (Signaling Transport) - протокол переноса сообщений сигнализации SS7 через IP-сеть.

Архитектура SIP (рис. 22) также очень проста и состоит из нескольких необязательных компонентов.

Рисунок 22 - Архитектура SIP




Клиент SIP (SIP user agent) - может быть представлен как устройством (IP-телефон, шлюз или другой пользовательский терминал), так и программным приложением для ПК, PDA и т. д. Обычно SIP-клиент содержит и клиентскую, и серверную часть (User Agent Client, или UAC, и User Agent Server, или UAS). Основные функции данного компонента - инициирование и завершение вызовов.

Прокси-сервер SIP - управляет маршрутизацией вызовов и работой приложения. Прокси-сервер не может инициировать или терминировать вызовы.

Redirect-сервер SIP - перенаправляет звонки согласно заданным условиям.

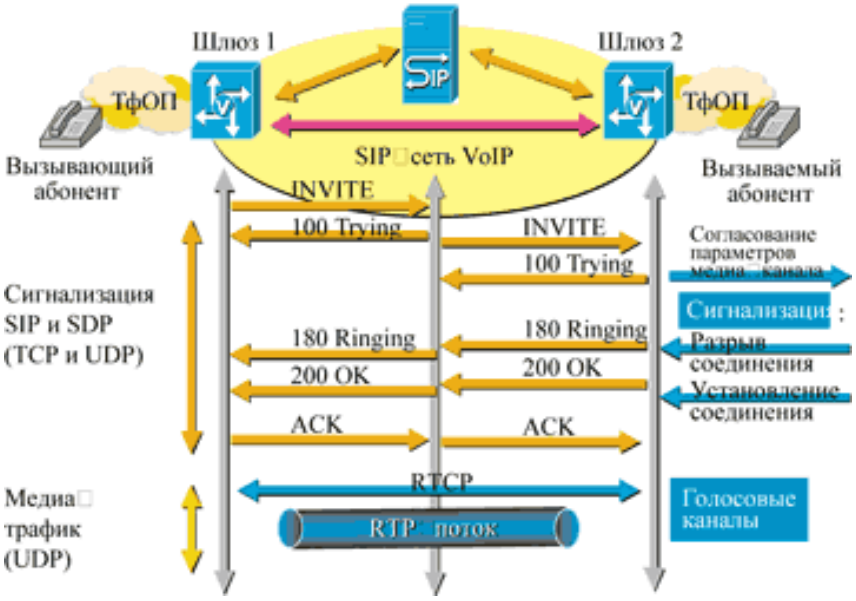
Сервер регистрации SIP (registrar/location) - осуществляет регистрацию пользователей и ведет базу соответствия имен пользователей их адресам, телефонным номерам и так далее.

Еще один важный компонент реальных SIP-сетей, хотя и не входящий формально в архитектуру SIP, - Back-to-Back User Agent (B2BUA). Это своеобразный сервер, представляющий собой два соединенных друг с другом SIP-клиента и поэтому способный инициировать и завершать вызовы.

В наиболее общей форме сценарий соединения по протоколу SIP с участием прокси-сервера показан на рис. 23. Абонент посылает на прокси-сервер запрос на соединение, отправляя сообщение Invite. Прокси-сервер возвращает сообщение Trying и передает сообщение Invite вызываемому абоненту. Вызываемая сторона отвечает сообщением Ringing, которое прокси-сервер пересылает вызывающей стороне. После того как вызываемый абонент снимет трубку, вызывающей стороне отправляется сообщение OK, которое

транслируется прокси-сервером. Вызываемому абоненту возвращается подтверждающее сообщение Ack.

Рисунок 23 - Сценарий соединения по протоколу SIP




С этого момента соединение считается установленным и начинается обмен медиа-трафиком по протоколам RTP/RTCP. Сторона, желающая завершить соединение, посылает сообщение Bye, и после получения подтверждающего OK соединение разрывается.

Этот сценарий очень прост, в нем не участвуют никакие другие серверы (Redirection, Registrar, Location), но он дает представление о схеме взаимодействия элементов SIP-сети.

**Протокол MGCP**

Последний из рассматриваемых протоколов IP-телефонии - MGCP (Media Gateway Control Protocol). Точнее, речь здесь идет не об одном протоколе, а о целой группе - SGCP, IPDC, MGCP, MEGACO, H.248. Эти спецификации не только очень схожи концептуально, но и являются "близкими родственниками".

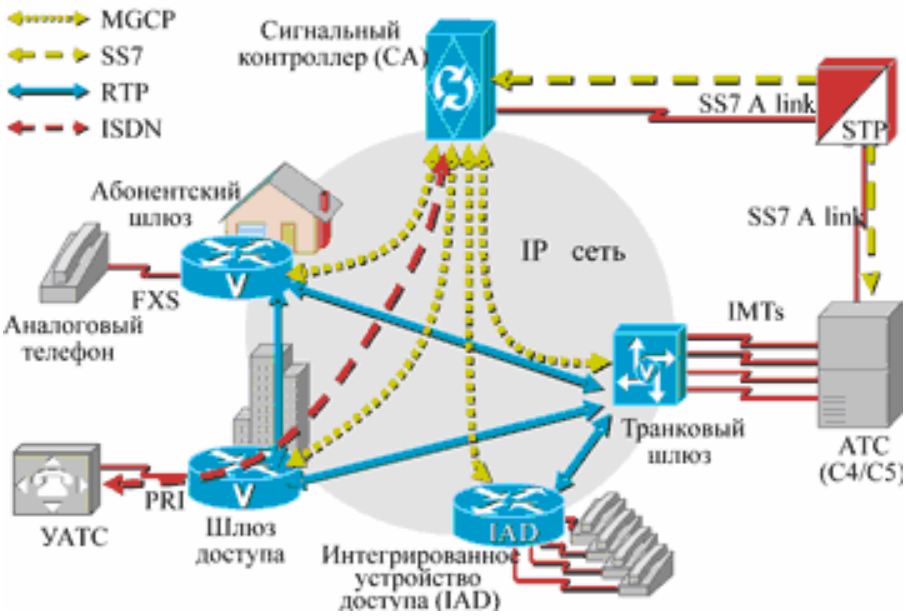
История формирования MGCP началась с создания двух протоколов - SGCP (Simple Gateway Control Protocol, разработка Bellcore и Cisco Systems) и IPDC (Internet Protocol for Device Control, разрабатывался компанией Level 3 при участии многих производителей). Затем SGCP и IPDC были объединены в один протокол, получивший название MGCP. В дальнейшем эволюция MGCP привела к появлению протоколов MEGACO (в рамках IETF) и H.248 (в рамках МСЭ).

Первая версия протокола MGCP (RFC 2705) датирована октябрем 1999 года. Интересно отметить, что MGCP - единственный из трех описываемых здесь протоколов, в работе над которым IETF и МСЭ сотрудничают; именно в результате этого взаимодействия и были созданы протоколы MEGACO и H.248. В то же время существуют и другие реализации MGCP-подобных протоколов, например, собственный протокол Cisco Systems SSCP (Skinny Station Control Protocol), с помощью которого УАТС Cisco Call Manager управляет IP-телефонами.

Основная идея MGCP состоит в том, что управление сигнализацией (Call Control) сосредоточено на центральном управляющем устройстве, называемом контроллером сигнализаций (Call Agent, CA), и полностью отделено от медиа-потоков (bearer). Эти потоки обрабатываются шлюзами или абонентскими терминалами, которые способны исполнять лишь ограниченный набор команд, исходящих от управляющего устройства. Архитектура

протокола MGCP-сети также очень проста (рис. 24), в ней выделяются всего два функциональных компонента. Первый может быть представлен шлюзом (Media Gateway, MG) или IP-телефоном, а второй - устройством управления вызовами, которое может называться контроллером сигнализаций (CA), контроллером шлюза (Media Gateway Controller, MGC) или программным контроллером (Softswitch, SS). Иногда контроллер сигнализаций представляют в виде двух компонентов - собственно контроллера (Call Agent), выполняющего функции управления шлюзами, и шлюза сигнализации (Signaling Gateway), обеспечивающего обмен сигнальной информацией и согласование между традиционной телефонной сетью и сетью IP.

Рисунок 24 - Архитектура MGCP



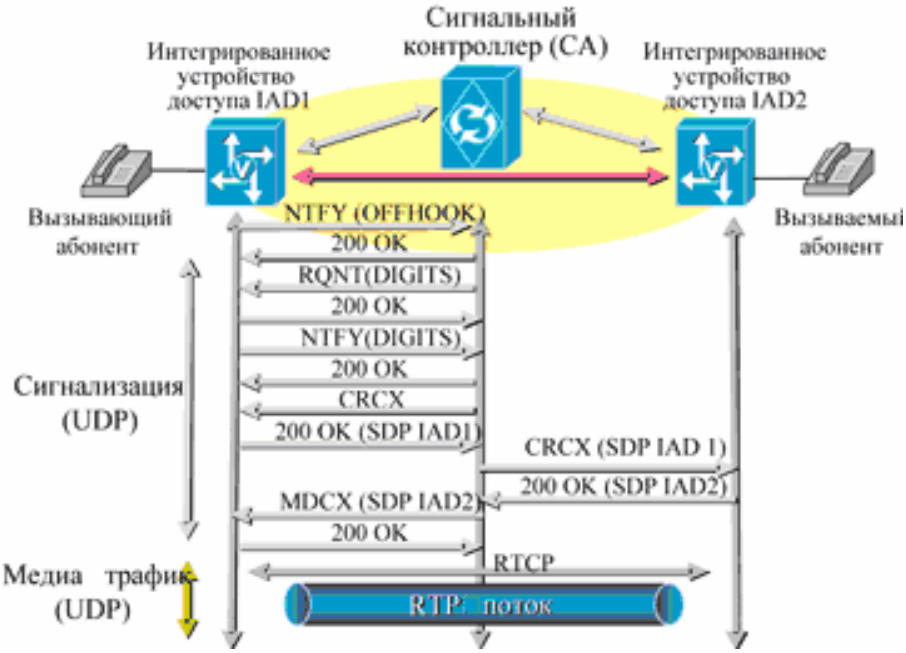

Контроллеры обмениваются со шлюзами (или IP-телефонами) данными в простом текстовом формате (в случае H.248 возможен и бинарный обмен), а функциональное назначение каждого шлюза определяется набором команд, которые он "понимает". Манипулируя наборами команд, можно получать специализированные шлюзы: транковые (Trunking gateways, TGW), абонентские (Residential gateways, RGW), шлюзы доступа (Access gateways, AGW) и так далее.

Контроллер сигнализаций CA воспринимает сеть как набор двух логических элементов - устройств (end-points) и соединений (connections) между ними. Устройства могут быть физическими (например, IP-телефоны или линии на шлюзах) или виртуальными (например, линии к серверам голосовых сообщений). Соединения могут быть ориентированы на передачу голоса, факс-сообщений или данных. Управление этими элементами, т. е. организация соединений между устройствами, происходит путем посылки команд в виде текстовых (ASCII) сообщений по протоколу UDP - при этом может использоваться протокол SDP. Как правило, управляющие воздействия контроллера CA инициируются какими-то событиями (events).

Простейший сценарий соединения в концепции MGCP (рис. 25) будет выглядеть следующим образом. Пользователь телефона, подключенного к MGCP-шлюзу, снимает трубку, после чего шлюз сообщает контроллеру об этом событии, а CA дает команду шлюзу включить в телефонную линию сигнал готовности (dial-tone). Теперь пользователь слышит в трубке непрерывный гудок. Набор телефонного номера - тоже последовательность событий для контроллера. Анализируя эти события, CA может установить соединение с другим абонентом в IP-сети или в телефонной сети. Централизованная обработка сигнализации дает возможность контроллеру прозрачно транслировать сигнализацию SS7 или ISDN из

телефонной сети в IP-сеть и, наоборот, получать соответствующие сигнальные сообщения, упакованные в IP-пакеты, а затем анализировать их и манипулировать голосовыми каналами на шлюзах.

Рисунок 25 - Сценарий соединения по протоколу MGCP




**Сравнение сигнальных протоколов**

Сравнивая особенности развития и функциональные особенности трех видов протоколов (таблица 4), можно сделать вывод, что их различия обусловлены историческими причинами, в частности, изменениями представлений о пути развития телекоммуникаций в разное время. При этом H.323 - это технологически устоявшийся, широко распространенный протокол IP-телефонии для операторских сетей и межоператорского обмена, "транзитный" протокол. В свою очередь, SIP - протокол предоставления расширенных голосовых услуг в IP-сетях, который продолжает быстро развиваться, "абонентский" протокол. Что касается MGCP, то он ориентирован, прежде всего, на организацию больших операторских узлов сопряжения IP-сетей с ТфОП и сетями SS7.

Таблица 4. Сравнение протоколов VoIP-сети

| Показатель | H.323 | SIP | MGCP |
|---|---|---|---|
| Клиент | Thick | Thick | Thin |
| Компонент, определяющий функциональность сети и сетевые сервисы | Привратник | Прокси-сервер | Сигнальный контроллер CA |
| Используемая модель | Телефонная (Q.931) | Интернет (WWW) | Централизованная |
| Протокол передачи сигнализации | TCP или UDP | TCP или UDP | UDP |
| Протокол передачи медиа-трафика | RTP | RTP | RTP |
| Формат сообщений | Двоичный (ASN.1) | Текстовый (ASCII) | Текстовый (ASCII) или двоичный |
| Стандартизирующая организация | ITU | IETF | IETF/ITU |

### 1.3.2. Протоколы передачи потоковых данных

Помимо протокола сигнализации другой важной составляющей частью IP телефонии является протокол RTP (Real Time Protocol), который обеспечивает сквозной сетевой транспорт для приложений требующих передачи потоковых данных в реальном времени, таких как аудио и видео.

Обычно, RTP работает поверх протокола UDP, используя функции мультиплексирования и проверки целостности данных данного протокола. Несмотря на то, что RTP очень часто используется для одноадресной передачи, в первую очередь он создавался для групповой (многоадресной) передачи. Помимо роли отправителя и получателя RTP определяет роли миксера и транслятора для поддержки многоадресных рассылок.

RTP является критическим компонентом VoIP, обеспечивая для получателя возможность переупорядочивания и хронометража пакетов перед воспроизведением. Заголовок RTP содержит временные отметки и последовательные номера, позволяющие буферизовать пакеты и устранять колебания длительности задержки. При потере пакета RTP не производит повторной передачи пакета.

Протокол RTCP отслеживает качество передачи данных и обеспечивает управляющую информацию. RTCP обеспечивает передачу информации следующим образом:

- для устройств вовлечённых в RTP сессию обеспечивается механизм обмена управляющей информацией и информацией о состоянии сессии. RTCP отслеживает такие показатели качества как: количество переданных и потерянных пакетов, задержка и колебание длительности задержки. RTCP использует определённый процент пропускной способности доступной для сессии, при этом управляющие пакеты передаются с интервалом не менее 5 секунд;
- RTCP передается отдельным потоком с использованием протокола UDP или TCP. При назначении номеров портов UDP для потока голосовых данных обычно выделяется чётный номер, в то время как для RTCP назначается следующий нечётный номер. Каждый голосовой звонок использует 4 порта – по 2 порта (один для RTP и один для RTCP) в каждом из направлений обмена данными.

### Протокол RTP

В Internet, также как и в некоторых других сетях, возможна потеря пакетов изменение их порядка в процессе транспортировки, а также вариация времени доставки в достаточно широких пределах. Мультимедийные приложения накладывают достаточно жесткие требования на транспортную среду. Для согласования таких требований с возможностями Интернет был разработан протокол RTP. Протокол RTP базируется на идеях, предложенных Кларком и Тенненхаузом [7], и предназначен для доставки данных в реальном масштабе времени (например, аудио или видео). При этом определяется тип поля данных, производится нумерация посылок, присвоение временных меток и мониторинг доставки. Приложения обычно используют RTP поверх протокола UDP для того, чтобы использовать его возможности мультиплексирования и контрольного суммирования. Но RTP может использоваться и поверх любой другой сетевой транспортной среды. RTP поддерживает одновременную доставку по многим адресам, если мультикастинг поддерживается нижележащим сетевым уровнем.

Следует иметь в виду, что сам по себе RTP не обеспечивает своевременной доставки и не предоставляет каких-либо гарантий уровня сервиса (QoS). Этот протокол не может гарантировать также корректного порядка доставки данных. Правильный порядок выкладки информации может быть обеспечен принимающей стороной с помощью порядковых номеров пакетов. Такая возможность крайне важна практически всегда, но особое внимание этому уделяется при восстановлении передаваемого изображения.

На практике протокол RTP не отделим от протокола RTCP (RTP control protocol). Последний служит для мониторинга качества обслуживания (QoS) и для передачи информации об участниках обмена в ходе сессии.

RTP гибкий протокол, который может доставить приложению нужную информацию, его функциональные модули не образуют отдельный слой, как правило, они встраиваются в прикладную программу. Протокол RTP не является жестко регламентирующим.

При организации аудио-конференции каждый участник должен иметь адрес и два порта, один для звуковых данных, другой для управляющих RTCP-пакетов. Эти параметры должны быть известны всем участникам конференции. При необходимости соблюдения конфиденциальности информация и пакеты управления могут быть зашифрованы. При аудио конференциях каждый из участников пересылает небольшие закодированные звуковые фрагменты длительностью порядка 20 мсек. Каждый из таких фрагментов помещается в поле данных RTP-пакета, который в свою очередь вкладывается в UDP-дейтограмму.

Заголовок пакета RTP определяет, какой вид кодирования звука применен (PCM, ADPCM или LPC), что позволяет отправителю при необходимости сменить метод кодирования, если к конференции подключился новый потребитель с определенными ограничениями или сеть требует снижения скорости передачи.

При передаче звука весьма важным становится взаимное положение закодированных фрагментов во времени. Для решения задачи корректного воспроизведения заголовки пакетов RTP содержат временную информацию и порядковые номера. Порядковые номера позволяют не только восстановить правильный порядок фрагментов, но и определить число потерянных пакетов-фрагментов.

Так как участники конференции могут появляться и исчезать по своему усмотрению, полезно знать, кто из них присутствует в сети в данный момент, и как до них доходят передаваемые данные. Для этой цели периодически каждый из участников транслирует через порт RTCP мультикастинг-сообщение, содержащее имя участника и диагностические данные. Узел-участник конференции шлет пакет BUY (RTCP), если он покидает сессию.

Если в ходе конференции передается не только звук, но и изображение, они передаются как два независимых потока с использованием двух пар UDP-портов. RTCP-пакеты посылаются независимо для каждой из этих двух сессий.

На уровне RTP не существует какой-либо взаимосвязи между аудио и видео сессиями. Только RTCP-пакеты несут в себе одни и те же канонические имена участников.

В некоторых случаях можно столкнуться с ситуацией, когда один из участников конференции подключен к сети через узкополосный канал. При этом нерационально требовать от всех участников перехода на кодировку, соответствующую этой малой полосе. Для того чтобы этого избежать, можно установить преобразователь, называемый *смесителем*, в непосредственной близости от узкополосной области.

Смеситель преобразует поток аудио-пакетов в последовательность пакетов, которая соответствует возможностям узкополосного канала. Эти пакеты могут быть одноадресными или мультикастными. Заголовок RTP включает в себя средства, которые позволяют мультиплексорам идентифицировать источники, внесшие вклад. Так что получатель может правильно идентифицировать источник звукового сигнала.

Некоторые участники конференции, использующие широкополосные каналы, не доступны для IP-мультикастинга (например, находятся за Firewall). Для таких узлов смесители не нужны, здесь используется другой RTP-уровень передачи, называемый *трансляцией*. Устанавливается два транслятора по одному с каждой из сторон Firewall. Внешний транслятор передает мультикастинг-пакеты по безопасному каналу внутреннему транслятору. Внутренний же транслятор рассылает их подписчикам локальной сети обычным образом.

Смесители и трансляторы могут выполнять и другие функции, например, преобразование IP/UDP пакетов в ST-II при видео конференциях.

Абсолютное время для протокола RTP представляется с помощью временных меток в соответствии с форматом NTP (network time protocol), который характеризует время в секундах от начала суток (UTC) 1 января 1900 [13]. Полное разрешение временной метки NTP определяется 64-битовым числом с фиксированной запятой без знака. Целочисленная часть задается первыми 32 битами, а дробная часть последними. В некоторых полях, где допустимо более компактное представление, используются только средние 32 бита (16 бит целочисленная часть и 16 бит дробная).

Заголовок RTP пакета имеет следующий формат (рис. 26).

Рисунок 26 - Формат заголовка RTP пакета


Первые 12 октетов присутствуют во всех RTP-пакетах, в то время как список CSRC-идентификаторов присутствует только, когда пакет формируется смесителем. Поля имеют следующие назначения:

**V (Версия):** 2 бита

Это поле идентифицирует версию протокола RTP. В настоящее время в это поле записывается код 2. Значение 1 использовалось в опытной версии RTP, а код 0 - в аудио приложении "vat".

**P (Заполнитель):** 1 бит

Если P=1, пакет содержит один или более дополнительных октетов-заполнителей в конце поля данных (заполнители не являются частью поля данных). Последний октет заполнителя содержит число октетов, которые должны игнорироваться. Заполнитель нужен при использовании некоторых алгоритмов шифрования при фиксированном размере блоков или при укладке нескольких RTP-пакетов в один UDP.

**X (Расширение):** 1 бит

Если бит X=1, далее следует фиксированный заголовок, за которым размещается одно расширение заголовка.

**CC (CSRC count - число CSRC):** 4 бита

Число CSRC содержит код количества CSRC-идентификаторов, которые записаны в пакете.

**M (маркер):** 1 бит

Интерпретация маркера определяется профайлом. Предполагается разрешить выделять в потоке пакетов существенные события, такие как границы кадра. Профайл может определить дополнительные маркерные биты или специфицировать отсутствие маркерных битов путем изменения числа битов в поле PT.

**PT(Тип данных):** 7 бит

Это поле идентифицирует формат поля данных RTP-пакета и определяет интерпретацию его приложением. Могут быть определены дополнительные коды типа данных. Исходный набор кодов по умолчанию для аудио и видео задан в профайле Internet-draft draft-ietf-avt-profile, и может быть расширен в следующих редакциях стандарта assigned numbers (RFC-1700) [20].

**Номер по порядку:** 16 бит

Номер по порядку инкрементируется на 1 при посылке очередного RTP-пакета данных, этот код может использоваться получателем для регистрации потерь пакетов и для восстановления истинного порядка присланных фрагментов. Начальное значение кода является случайным. Алгоритм генерации таких кодов рассмотрен в [9].

**Временная метка:** 32 бита

Временная метка соответствует времени стробирования для первого октета в информационном RTP-пакете. Время стробирования должно быть получено от часов, показания которых увеличиваются монотонно и линейно, чтобы обеспечить синхронизацию и вычисление временного разброса. Разрешающая способность часов должна быть достаточной для обеспечения приемлемой точности синхронизации (одного тика на видео кадр обычно не достаточно). Частота часов зависит от формата данных и задается статически в профайле, в спецификации поля данных, или динамически средствами, выходящими за пределы спецификации протокола RTP. Если RTP-пакеты генерируются периодически, используется временная привязка, определенная задающим генератором стробирования, а не показаниями системных часов.

Начальное значение временной метки является случайным. Несколько последовательных RTP-пакетов могут иметь идентичные временные метки, если логически они генерируются одновременно (например, относятся к и тому же видео кадру).

**SSRC:** 32 бита

Источник потока RTP-пакетов, определяется 32-битным числовым SSRC-идентификатором, который записывается в заголовок RTP-пакета и не зависит от сетевого адреса. Все пакеты от источника синхронизации образуют часть с идентичной временной привязкой и нумерацией. Эти данные используются принимающей стороной при воспроизведении. Источниками синхронизации могут служить источники первичного сигнала (микрофоны или видеокамеры), а также RTP-смесители. SSRC-идентификатор представляет собой случайное число, которое является уникальным для данной RTP-сессии. Участник сессии не должен использовать один и тот же SSRC-идентификатор для всех RTP-сессий мультимедийного набора. Если участник формирует несколько потоков в рамках одной RTP-сессии (например, от нескольких видеокамер), каждый участник должен быть снабжен уникальным SSRC-идентификатором. Этот идентификатор выбирается случайным образом, так чтобы в пределах одной RTP-сессии не было двух равных SSRC-кодов. Все приложения должны быть способны выявлять случаи равенства SSRC-кодов. Если отправитель изменяет свой транспортный адрес, он должен также сменить и SSRC-идентификатор.

**CSRC-список:** от 0 до 15 элементов, по 32 бита каждый

CSRC-список идентифицирует источники информации, которые внесли свой вклад в поле данных пакета. Это позволяет принимающей стороне идентифицировать передающего, хотя все пакеты имеют один и тот же SSRC-идентификатор. Число идентификаторов задается полем *CC*. Если число источников больше 15, только 15 из них могут быть идентифицированы.

Хотя существующие RTP-заголовки позволяют решать широкий круг проблем, предусмотрена возможность их модификации с помощью профайлов. При этом сохраняется возможность контроля и мониторинга с использованием стандартных средств.

Бит маркера и поле типа данных содержат информацию, задаваемую профайлом, но они размещаются в стандартном заголовке, так как нужны многим приложениям. Октет, где размещаются эти поля, может быть переопределен профайлом. При любом числе маркерных битов один должен размещаться в старшем разряде октета, так как это необходимо для мониторинга потока;

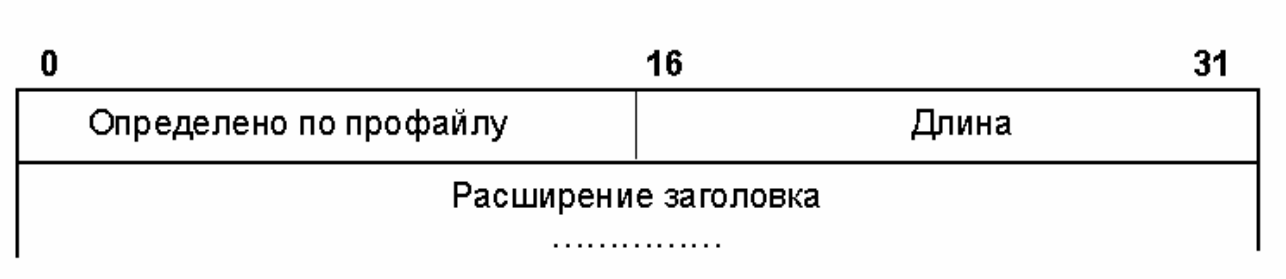
Дополнительная информация, которая необходима для конкретного формата поля данных, такая как тип видео-кодирования, должна транспортироваться в поле данных пакета. Это может быть заголовок, присутствующий в начале поля данных;

Если конкретное приложение нуждается в дополнительных возможностях, которые не зависят от содержимого поля данных, профайл данного приложения должен определить дополнительные фиксированные поля, следующие непосредственно после поля SSRC существующего заголовка пакета. Эти приложения смогут получить доступ к этим дополнительным полям, при этом сохраняются все стандартные средства контроля и мониторинга, так как они базируются на первых 12 октетах заголовка.

В протоколе RTP предусмотрен механизм расширений заголовка, который позволяет модифицировать заголовок и экспериментировать с новыми форматами поля данных. Этот механизм устроен так, что расширения заголовка могут игнорироваться приложениями, которые не нуждаются в расширениях.

Расширения заголовка предназначены для ограниченного использования. Большинство приложений предпочтительно реализовать, используя профайл. Формат реализации расширений показан на рис. 27.

Рисунок 27 - Формат расширения заголовка RTP пакета




Если бит X в RTP заголовке равен 1, то к заголовку добавлено расширение переменной длины, за которым может следовать список CSRC. Расширение заголовка содержит 16-битовое поле длины, определяющее число 32-битных слов в расширении, исключая 4-октета заголовка расширения (таким образом, значения поля длина, равное нулю вполне допустимо). Информационный заголовок RTP может иметь только одно расширение. Для того чтобы обеспечить работу различных приложений с различными расширениями заголовка или чтобы обеспечить работу с более чем одним типом расширений, первые 16 бит расширения заголовка остаются свободными для выбора идентификаторов или параметров. Формат этих 16 бит определяется спецификацией профайла, с которым работает приложение.

Кроме оконечных систем RTP поддерживает трансляторы и смесители, которые рассматриваются как промежуточные системы на уровне RTP.

RTP транслятор/смеситель соединяет две или более области на транспортном уровне. Обычно каждая область определяется сетью и транспортным протоколом (например, IP/UDP), групповым адресом или парой индивидуальных адресов, а также портом назначения транспортного уровня. Одна система может служить транслятором или смесителем для нескольких RTP сессий.

Существует большое разнообразие трансляторов и смесителей, спроектированных для решения различных задач и приложений. Некоторые служат для шифрования/дешифрования несущих дейтограмм. Различие между смесителями и трансляторами заключается в том, что последние пропускают через себя потоки данных, при необходимости их преобразуя, а смесители объединяют несколько потоков в один.

**Транслятор.** Переадресует RTP-пакеты, не изменяя их SSRC-идентификаторы. Это позволяет получателям идентифицировать отдельные источники, даже если пакеты от всех источников проходят через один общий транслятор и имеют сетевой адрес транслятора. Некоторые типы трансляторов передают данные без изменений, другие кодируют данные и соответственно изменяют коды типа данных и временные метки. Приемник не может заметить присутствия транслятора.

**Смеситель.** Принимает потоки RTP-данных от одного или нескольких источников, может изменять формат данных, определенным образом объединяет потоки и затем формирует из них один общий поток. Так как объединяемые потоки не синхронизованы, смеситель производит синхронизацию потоков и формирует свою собственную временную шкалу для исходящего потока. Смеситель является источником синхронизации. Таким образом, все пакеты данных, переадресованные смесителем, будут помечены SSRC-идентификатором смесителя. Для того чтобы сохранить информацию об источниках исходных данных, смеситель должен внести свои SSRC-идентификаторы в список CSRC-идентификаторов, который следует за фиксированным RTP-заголовком пакета. Смеситель, который, кроме того, вносит в общий поток свою составляющую, должен включить свой собственный SSRC-идентификатор в CSRC-список для данного пакета.

Для некоторых приложений смеситель может не идентифицировать источники в CSRC-списке. Однако это создает опасность того, что петли, включающие эти источники, не смогут быть выявлены.

Преимуществом смесителя перед транслятором для аудио-приложений является то, что выходная полоса не превосходит полосы одного источника, даже когда в сессии на входе смесителя присутствуют несколько участников. Недостатком является то, что получатели с выходной стороны не имеют никаких средств для контроля того, какой из источников передает данные даже в случае наличия дистанционного управления смесителем.

В рамках RTP-стандарта определены следующие элементы поля данных (этот список не следует рассматривать, как окончательный):

**Заголовок поля данных RTP.** Октет RTP заголовка, содержащий маркер тип поля данных, может быть переопределен с помощью профайла (например, можно изменить число маркерных битов).

**Типы поля данных.** Профайл обычно определяет набор форматов поля данных (напр., типов кодирования исходных данных) и соответствие между этими форматами и кодами типа поля данных. Для каждого описанного типа поля данных должна быть определена частота временных меток.

**Дополнения к заголовку RTP.** К стандартному RTP заголовку могут быть добавлены новые поля, расширяющие функциональность приложения.

**Расширения заголовка RTP.** Структура содержимого первых 16 бит расширения RTP заголовка должна быть определена профайлом.

Безопасность. Профайл может специфицировать, какие услуги и алгоритмы безопасности должно обеспечить приложение.

**Установка соответствия между строкой и ключом.** Профайл может специфицировать, какому ключу шифрования соответствует введенный пользователем пароль.

**Нижележащий протокол.** Определяется нижележащий транспортный протокол, который служит для пересылки RTP пакетов.

**Транспортное соответствие.** Соответствие RTP и RTCP адресам транспортного уровня, например, UDP-портам.

**Инкапсуляция.** Инкапсуляция RTP-пакетов может быть определена для того, чтобы позволить транспортировку нескольких RTP-пакетов в одной дейтограмме нижележащего протокола.

Не предполагается, что для каждого приложения требуется свой профайл. В пределах одного класса приложений целесообразно использовать расширения одного и того же профайла. Простое расширение, такое как введение дополнительного типа поля данных или нового типа RTCP-пакета, может быть выполнено путем регистрации их через комитет по стандартным числам Интернет и публикации их описаний в приложении к профайлу.

**Протокол RTCP**

Управляющий протокол RTCP (RTP control protocol) базируется на периодической передаче управляющих пакетов всем участникам сессии, используя тот же механизм рассылки, что и для пакетов данных. Этот протокол не имеет самостоятельного значения и используется лишь совместно с RTP. Нижележащий протокол должен обеспечивать мультиплексирование пакетов данных и управления, используя разные номера портов. RTCP выполняет четыре функции:

Главной задачей данного протокола является обеспечение обратной связи для контроля качества при рассылке данных. Обратная связь может быть непосредственно полезна при адаптивном кодировании [4,11], но эксперименты с IP мультикастингом показали, что для получателей крайне важно диагностировать ошибки при рассылке пакетов. Посылка сообщений-отчетов о приеме данных всем участникам позволяет тому, кто обнаружил какие-то проблемы, разобраться в том, являются ли эти трудности локальными или глобальными. При механизме рассылки типа IP-мультикастинга, сервис провайдер, который непосредственно не вовлечен в сессию, получив обратную связь, может независимо отслеживать ситуацию в сети.

RTCP имеет постоянный идентификатор транспортного уровня для RTP источника, который называется каноническим именем или Cname. Так как SSRC-идентификатор может быть изменен, если будет зафиксировано столкновение или источник будет вынужден рестартовать, получатели нуждаются в Cname, для того чтобы отслеживать каждого из участников. Получателям также нужно Cname, чтобы установить соответствие между многими потоками данных от одного участника при реализации нескольких сессий одновременно, например, чтобы синхронизовать аудио- и видео-каналы.

Стандарт определяет несколько типов RTCP пакетов, которые предназначены для переноса управляющей информации:

**SR:** Отчет отправителя. Для статистики приема и передачи участников, которые являются активными отправителями

**RR:** Отчет получателя. Для получения статистики от участников, которые не являются активными отправителями

**SDES:** Элементы описания источника, включая cname

**BYE:** Отмечает прекращение участия в группе

**APP:** Специфические функции приложения

Каждый RTCP пакет начинается с фиксированной части, сходной с той, которая используется RTP-пакетами, за ней следуют структурные элементы, которые могут иметь переменную длину в зависимости от типа пакета, но кратную 32 бит. Несколько RTCP пакетов могут быть соединены друг с другом без введения каких-либо сепараторов, для того чтобы получить составной RTCP пакет, который посылается в рамках транспортного протокола низкого уровня, например UDP. Не существует специального счетчика индивидуальных RTCP пакетов, так как протокол нижнего уровня задаст общую длину и определит конец составного пакета.

Каждый индивидуальный RTCP пакет в составном пакете может обрабатываться независимо без каких-либо требований к порядку или комбинации пакетов. Однако, для того чтобы выполнить функции протокола накладываются следующие ограничения:

- Статистика приема (в SR или RR) должна посылаться так часто, как это позволяют ограничения пропускной способности, так что каждый периодически посылаемый составной пакет включает в себя пакет отчета;
- Новые получатели должны приобрести Cname для источника как можно быстрее, каждый составной RTCP-пакет должен включать в себя SDES Cname;
- Число типов пакетов, которые могут впервые появиться в составном пакете, должно быть ограничено.

Таким образом, все RTCP пакеты должны посылаться в составных пакетах (не менее 2) и иметь следующий рекомендованный формат:

**Префикс шифрования.** Если составной пакет должен быть зашифрован, он снабжается 32-битным случайным числом-префиксом, которое копируется для каждого передаваемого составного пакета.

**SR или RR.** Первый RTCP-пакет в составном пакете должен быть всегда сообщением-отчетом. Это справедливо, даже если не было послано или получено никаких данных, в этом случае посылается пустой пакет RR. Это справедливо, даже если другим RTCP пакетом в составной дейтаграмме является Bye.

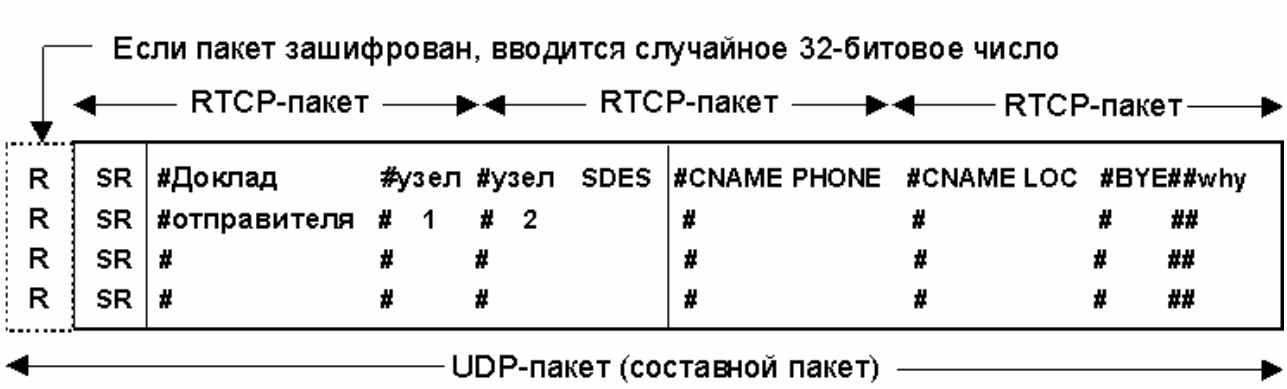
**Дополнительные RR.** Если число источников, для которых приводится статистика приема, превышает 31, в первый пакет помещается информация по части источников, остальная часть размещается в следующих RR-пакетах.

**SDES.** SDES-пакет, содержащий Cname, должен быть включен в каждый составной RTCP-пакет. Другие элементы описания источника могут быть опционально добавлены, если этого требует характер приложения и позволяет пропускная способность используемого канала.

**BYE или APP.** Другие типы RTCP-пакетов, включая те, которые еще предстоит определить, могут следовать далее в произвольном порядке. Пакет Bye, если он присутствует, должен быть последним и содержать SSRC/CSRC. Пакеты одного и того же типа могут повторяться.

Для трансляторов и смесителей рекомендуется объединять RTCP-пакеты от нескольких источников. Пример составного RTCP пакета, который может быть сформирован смесителем, представлен на рис. 28. Если полная длина составного пакета превысит максимальный размер пересылаемого блока данных для сети (MTU), он может быть фрагментирован и переслан в нескольких составных пакетах нижележащего транспортного протокола. Следует отметить, что каждый составной пакет должен начинаться с SR или RR-пакета.

Рисунок 28 - Пример составного пакета RTCP




Приложение может игнорировать RTCP пакеты неизвестного ему типа. Дополнительные типы RTCP-пакетов могут быть зарегистрированы IANA (Internet Assigned Numbers Authority).

Протокол RTP построен так, чтобы позволять приложению изменять число участников от единиц до тысяч. Например, при аудио конференциях информационный поток всегда ограничен (сколько бы не было участников, все они одновременно говорить не могут). Однако, трафик управления таким свойством не обладает. Если доклады о приеме от каждого участника поступают с постоянной частотой, трафик управления будет расти пропорционально числу участников. Следовательно, нужно принимать меры по ограничению трафика.

Для каждой сессии предполагается, что предельно допустимый информационный трафик сессии делится между участниками. Эта полоса пропускания может быть

зарезервирована. Полоса не зависит от метода кодирования, но на выбор метода кодирования может оказать влияние имеющаяся в распоряжении полоса пропускания используемого канала. Определенные ограничения на полосу сессии может накладывать конкретное приложение. Вычисления полосы пропускания, необходимой для управления, требует учета издержек транспортных протоколов (например, UDP и IP).

Трафик управления должен быть ограничен малой долей полной полосы пропускания сессии: настолько малой, чтобы не нанести ущерба основной функции транспортного протокола - переносу информации. Предлагается, чтобы доля трафика сессии, выделенная на RTCP была фиксирована на уровне не более 5%. Параметры, определяющие трафик, должны быть идентичными для всех участников, так чтобы они могли корректно вычислить период рассылки отчетов.

Данная спецификация определяет несколько элементов описания источника (SDES). Сюда входит CNAME (каноническое имя), Name (персональное имя) и Email (электронный адрес). Спецификация предлагает также средства для определения типа RTCP-пакетов, специфического для конкретного приложения. Приложения должно проявлять определенную осторожность при выделении полосы для любой дополнительной информации, так как это неизбежно вызовет замедление скорости предоставления отчетов и задержит присылку. Рекомендуется, чтобы дополнительная информация индивидуального участника не занимала более 20% полосы, выделенной для RTCP. Более того, даже не предполагается, что все элементы SDES будут включаться каждым приложением. Например, приложение может посылать только CNAME, Name и Email и не посылать более никакой дополнительной информации. Name может быть присвоен более высокий приоритет чем Email, так как Name будет отображаться пользовательским интерфейсом приложения постоянно, в то время как Email может отображаться только при запросе. При каждой RTCP рассылке, должны посылаться RR- и SDES-пакеты. Последний содержит элемент Cname. Для небольших сессий, работающих с минимальными периодами рассылки, это будет делаться в среднем каждые 5 секунд. Каждая третья рассылка (15 секунд) может содержать один дополнительный элемент в пакете SDES. Семь из восьми раз это будет элемент Name, и каждый восьмой раз (2 минуты) это будет элемент Email.

Когда несколько приложений работают одновременно, например, в случае мультимедиа конференции, допускается, чтобы дополнительная информация пересылалась только в рамках одной RTP-сессии. Остальные сессии будут использовать только элемент Cname.

RTP-получатели обеспечивают обратную связь контроля качества, используя RTCP пакеты отчетов, которые могут принимать ту или иную форму в зависимости от того, является ли получатель одновременно и отправителем. Единственным различием между формами отчета отправителя (SR) и получателя (RR), помимо кода типа пакета, является то, что отчет отправителя содержит 20-байтовую секцию информации об отправителе. SR посылается, если узел отправил какие-либо информационные пакеты за время подотчетного периода (с момента отправки предыдущего отчета), в противном случае отправляется пакет RR.

Как SR так и RR формы включают в себя нуль или более блоков отчетов о приеме, один для каждого источника синхронизации, от которого получатель принял информационные RTP-пакеты с момента последнего отчета. Отчеты не направляются для источников, перечисленных в списке CSRC. Каждый блок отчета о приеме содержит статистику данных, полученных от конкретного источника. Так как в SR или RR-пакет можно поместить максимум 31 блок отчетов, дополнительные RR-пакеты укладываются после исходного SR или RR-пакета.

Когда нет информации об отправке или приеме, в начало составного RTCP пакета вставляется пустой RR-пакет (RC = 0).

Профайл должен определять специфические для приложения расширения в докладах получателей и отправителей, если имеется дополнительная информация о получателе или

отправителе, которая должна регулярно сообщаться. Этот метод предпочтительнее, чем описание нового типа RTCP-пакета, так как не требует дополнительных издержек.

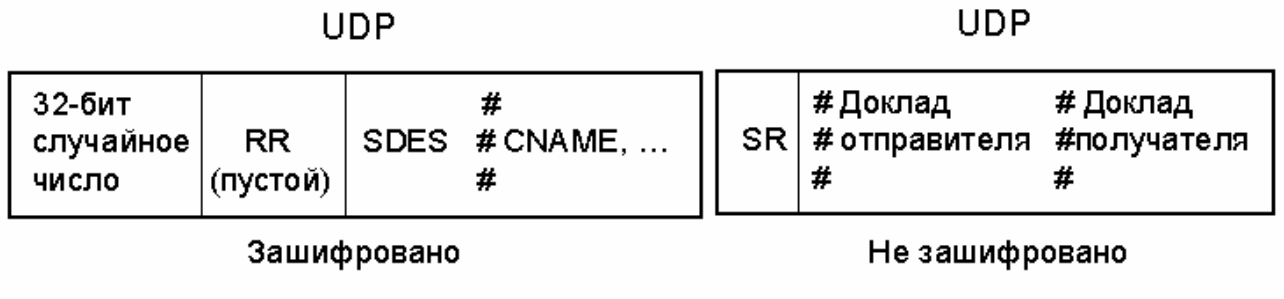
Если необходима дополнительная информация, она должна быть включена в первую очередь в расширение для отчета отправителя, но не будет присутствовать в отчетах о приеме. Если должна быть подключена информация о получателях, эти данные могут структурироваться в виде массива блоков дополнительно к существующему массиву блоков-отчетов, то есть, число блоков будет задано полем RC.

Ожидается, что качество обратной связи важно не только для отправителя и получателей, но и для независимых мониторов. Отправитель может модифицировать свою передачу на основе обратной связи, получатели могут определить, являются ли проблемы локальными, региональными или глобальными. Менеджер сети может использовать независимые мониторы, которые получают только RTCP-пакеты, а не соответствующие информационные RTP-пакеты, для оценки эксплуатационных параметров своей сети для многоадресного обмена.

На основе информации отправителя независимый монитор может вычислить усредненное значение потока данных, не получая этих данных. Если можно предположить независимость вероятности потери пакета от его размера, тогда число полученных пакетов, умноженное на средний размер поля данных, может дать оценку для пропускной способности получателя.

Для RTCP допустимо расщепление составных пакетов на пакеты нижележащего уровня, один зашифрованный и один открытый. Например, информация SDES может быть зашифрована, в то время как отчеты о приеме будут посылаться открыто для обеспечения мониторинга. В примере, представленном на рис. 29 информация SDES должна быть присоединена к RR-пакету, не содержащему отчет. Таким образом, соблюдается правило о том, что все составные пакеты начинаются с SR или RR пакетов.

Рисунок 29 - Зашифрованный и незашифрованный RTCP-пакеты




Пакет SDES состоит из заголовка и нескольких фрагментов (рис. 30), каждый из которых содержит элементы описания источника, соответствующего данному фрагменту (число фрагментов может быть равно нулю).

Рисунок 30 - Формат пакета SDES (RTCP-пакет описания источника)


Поля версия (V), заполнитель (P) и длина имеют то же назначения что и в случае SR-пакетов.

**Тип пакета (PT):** 8 бит

Содержит константу 202, которая идентифицирует данный пакет как RTCP SDES.

**Число источников (SC):** 5 бит

Число фрагментов SSRC/CSRC, содержащихся в данном SDES-пакете. Значение нуль допустимо, но бесполезно.

Каждый фрагмент состоит из идентификатора SSRC/CSRC, за которым следует список элементов описания источника SSRC/CSRC (число элементов может равняться нулю). Каждый фрагмент начинается на 32-битовой границе. Каждый элемент состоит из 8-битового поля типа, 8-битового поля числа октетов, характеризующего длину текста, исключая эти 2 октета заголовка, и собственно текста. Заметьте, что текст не может содержать более 255 октетов, но это вполне согласуется с требованиями ограничений на полосу, выделяемую для RTCP-пакетов.

Текст кодируется согласно требованиям UTF-2, описанным в стандарте 10646 [21,12], annex F ISO. Эта кодировка известна также под названием UTF-8 или UTF-FSS. Она описана в документе "File System Safe UCS Transformation Format (FSS_UTF)", "X/open preliminary specification", документ номер P316 и "Unicode Technical Report #4". US-ASCII являются модификациями данного кодирования и требуют определенных доработок. Присутствие многооктетного кодирования задается путем установления старшего бита октета символа равным 1.

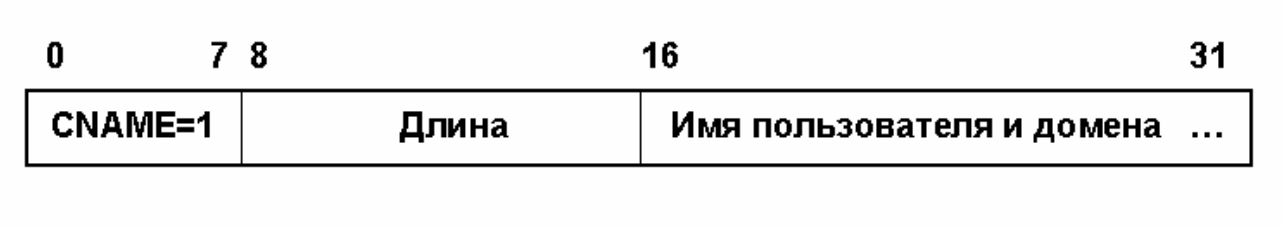
Описания элементов плотно прилегают друг к другу, то есть, их описания не выравниваются на 32-битовые границы путем индивидуального заполнения. Текст не завершается нулем, так как мультиоктетное кодирование может включать в себя нули. Список элементов в каждом фрагменте завершается одним или несколькими нулевыми октетами, первый из которых интерпретируется как тип элемента нуль, завершающий список, а последующие служат для заполнения до 32-битовой границы. Фрагменты, содержащие только нулевые элементы (4 нулевых октета), допускаются, но бесполезны.

Оконечные системы посылают один пакет SDES, содержащий их собственный идентификатор источника (то же, что и SSRC в фиксированных RTP-заголовках). Смеситель посылает один пакет SDES, содержащий фрагмент для каждого источника, от которого поступает SDES-информация, или несколько SDES-пакетов описанного выше формата в случае, когда число таких источников больше 31.

Из числа SDES-элементов только Cname (рис. 31) является обязательным. Некоторые элементы, описанные ниже, могут оказаться полезными только для определенных профайлов, но типы элементов выделяются из общего кодового пространства, с тем чтобы

обеспечить совместную работу различных приложений. Дополнительные элементы могут быть определены в профайле путем регистрации их кодов IANA.

Рисунок 31 - Формат Cname (канонический идентификатор конечной системы)




Идентификатор Cname имеет следующие свойства:

- Так как характеризуемый случайным числом идентификатор SSRC может измениться, если обнаруживается конфликт или если программа перезапускается, элемент Cname должен обеспечить связь между идентификатором SSRC и источником, которая должна оставаться неизменной;
- Подобно идентификатору SSRC, идентификатор Cname должен быть уникальным для каждого из участников RTP-сессии;
- Чтобы обеспечить связь между мультимедийными средствами, используемыми одним и тем же участником в наборе взаимосвязанных RTP-сессий, Cname должно быть зафиксировано для данного участника;
- Для того чтобы обеспечить независимый мониторинг, Cname должно быть удобным средством идентификации источника как для программы, так и для человека.

Следовательно, Cname должно по возможности получаться алгоритмически, а не вводиться вручную. Чтобы удовлетворить этому требованию следует использовать описанный ниже формат, если другой синтаксис или семантика не задана. Элемент Cname должен иметь формат "user@host", или "host", если имя пользователя не доступно, как это бывает в однопользовательских системах. Для обоих форматов, "host" является либо полным именем домена ЭВМ, откуда поступают данные в реальном масштабе времени, форматированные согласно требованиям документов RFC-1034 [14], RFC-1035 [15] и раздела 2.1 RFC-1123 [3]; или стандартным ASCII-представлением цифрового, сетевого адреса интерфейса ЭВМ, используемого для RTP-обмена. Например, стандартное ASCII-представление IP-адреса (версия 4) в "точечно-цифровом" виде. Стандартное полное имя домена более удобно для человека и исключает необходимость посылать в дополнение элемент Name, но в некоторых обстоятельствах его может быть трудно или невозможно получить. Примерами могут служить "dwarf@sleepy.beauty.com" или "dwarf@192.166.148.9" для мультипользовательских систем. В системах, где нельзя получить имя пользователя, можно применить "sleepy.beauty.com" или "192.166.148.9".

Имя пользователя должно иметь форму, которая может быть использована в запросах "Finger" или "Talk", то есть, это скорее имя, вводимое при аутентификации, чем истинное имя пользователя. Имя ЭВМ не обязательно идентично электронному почтовому адресу участника.

Этот синтаксис не обеспечит уникальность имени в тех случаях, когда приложение позволяет пользователю сформировать несколько источников на своей ЭВМ. Такое приложение должно полагаться на SSRC для дополнительной идентификации источника, или на профайл, для которого приложение должно будет специфицировать синтаксис идентификаторов Cname.

Если каждое приложение создает свои Cname независимо, в результате можно получить дублирующие имена. Если необходимо осуществить связь между сессиями, работающими в разных средах, должны быть использованы специальные средства, которые с одной стороны обеспечат уникальность имен, а с другой припишут идентичные имена источникам, размещенным в одной ЭВМ, но работающих с разными средами.

Дубликаты имен могут возникать, когда ЭВМ с частными адресами [19], не имеющие выхода в Интернет, переадресуют свои RTP-пакеты в Интернет через транслятор RTP-уровня. Для того чтобы разрешать такие конфликты приложение должно иметь средства для выработки и присвоения уникальных имен Cname.

Элемент Name - это настоящее имя, используемое для описания источника. Оно может быть сформировано пользователем в произвольной форме. Для приложений типа конференций эта форма имени может быть наиболее желательной при отображении в списках участников и, следовательно, может посылаться более часто, чем любые другие элементы помимо Cname. Такой приоритет может быть установлен профайлом. Значение name предполагается неизменным, по крайней мере, в пределах сессии. В то же время не требуется, чтобы оно было уникальным для группы участников сессии.

Адрес электронной почты должен иметь формат, согласующийся с требованиями документа RFC-822 [8]. Значение элемента Email предполагается неизменным в пределах сессии.

Телефонный номер должен иметь формат с символом плюс, замещающим международный код. Например, "+7 495 123 4567" для номера в России.

Географическое положение узла. Различная детализация этого элемента сильно зависит от приложения. Значение LOC предполагается неизменным на время сессии. Исключение могут составлять мобильные ЭВМ.

Строка, сообщающая имя и, возможно, версию приложения, формирующего поток, например, "VC 2.1". Эта информация может быть полезной для отладочных целей и сходна с SMTP-заголовками. Предполагается, что значение TOOL остается постоянным в течение сессии.

Элемент Note предназначен для сообщений, характеризующих текущее состояние источника, например, "on the phone, can't talk". Или, во время семинара этот элемент может быть использован для передачи темы обсуждения. Он может служить только для передачи необычной информации и не должен включаться в систематическую рассылку, так как замедлит скорость передачи отчетов. В частности, он не должен включаться в конфигурационный файл пользователя.

Так как может быть важно отобразить элемент Note (в случае, когда он активен), скорость, с которой передаются другие элементы (кроме Cname), такие как Name, может быть уменьшена с тем, чтобы передать элемент Note. Когда сообщение становится не актуальным, элемент Note передается еще несколько раз с той же частотой, но с длиной строки, равной нулю. Однако, получатели должны рассматривать элемент Note как потерявший актуальность, если они не получают его, например, на протяжении 20-30 RTCP-интервалов.

Элемент частного расширения SDES (рис. 32) используется, для того чтобы определить экспериментальные или специфические для приложения расширения SDES. Элемент содержит префикс, включающий в себя субполя длины и строки префикса, за которыми следует строка значения, занимающая остальное пространство элемента, и несущая необходимую информацию. Поле длины префикса занимает 8 бит. Строка префикса представляет собой имя, определенное человеком, который сформировал элемент PRIV. Это имя должно быть уникальным и никакой другой элемент PRIV не может иметь такое же. Разработчик приложения может выбрать имя приложения плюс, если необходимо, дополнение.

Рисунок 32 - Формат элемента расширения PRIV (элемент частного расширения SDES)


Следует заметить, что префикс занимает некоторое место, из числа 255 октетов элемента, по этой причине желательно, чтобы он был короче.

Префиксы SDES PRIV не нужно регистрировать в IANA. Если некоторая форма элемента PRIV окажется достаточно универсальной, она должна быть приписана некоторому регулярному типу элемента SDES, зарегистрированному IANA, так что необходимость в префиксе отпадет. Это упростит использование и увеличит эффективность передачи.

Значения типов пакетов RTCP (таблица 5) были выбраны в диапазоне 200-204 для улучшенного контроля корректности заголовков RTCP пакетов. Когда поле типа пакета RTCP сравнивается с соответствующим октетом RTP-заголовка, этот диапазон соответствует маркерному биту 1 (который обычно отсутствует в информационных пакетах) и старшему биту стандартного поля типа данных равному 1 (так как статические типы поля данных обычно лежат в младшей половине).

Таблица 5. Типы пакетов RTCP

| Сокращенное название | Имя | Значение |
|---|---|---|
| SR | sender report - сообщение отправителя | 200 |
| RR | receiver report - сообщение получателя | 201 |
| SDES | source description - описание источника | 202 |
| BYE | goodbye - завершение | 203 |
| APP | application-defined - определен приложением | 204 |

Другие константы определены IANA. Экспериментаторам предлагается зарегистрировать числа, которые им нужны, а затем аннулировать регистрацию, если необходимость в них отпадет.

# 2. Диагностика и анализ локальных сетей

Очень часто под диагностикой локальной сети подразумевают тестирование только ее кабельной системы. Это не совсем верно. Кабельная система является одной из важнейших составляющих локальной сети, но далеко не единственной и не самой сложной с точки зрения диагностики. Помимо состояния кабельной системы на качество работы сети значительное влияние оказывает состояние активного оборудования (сетевых плат, коммутаторов и маршрутизаторов), качество оборудования сервера и настройки сетевой операционной системы. Кроме того, функционирование сети существенно зависит от алгоритмов работы эксплуатируемого в ней прикладного программного обеспечения.

Под термином "сеть" здесь подразумевается весь комплекс указанных выше аппаратных и программных средств; а под термином "диагностика сети" - процесс определения причин неудовлетворительной работы прикладного программного обеспечения (ПО) в данной сети. Именно качество работы прикладного ПО в сети оказывается определяющим с точки зрения пользователей [1]. Все прочие критерии, такие как число ошибок передачи данных, степень загруженности сетевых ресурсов, производительность оборудования и тому подобное, являются вторичными.

## 2.1. Методика диагностики сети

Основных причин неудовлетворительной работы прикладного ПО в сети может быть несколько: повреждения кабельной системы, дефекты активного оборудования, перегруженность сетевых ресурсов (канала связи и сервера), ошибки самого прикладного ПО. Часто одни дефекты сети маскируют другие. Таким образом, чтобы достоверно определить, в чем причина неудовлетворительной работы прикладного ПО, локальную сеть требуется подвергнуть комплексной диагностике.

При возникновении неполадок работы сети поиск неисправности и ее устранение происходит в строгом соответствии с семиуровневой моделью сети ISO OSI. Последовательно проверяются на наличие ошибок уровни начиная с физического, после проверки каждого уровня проверяется вышележащий.

### Организация диагностики локальной сети

В рамках предлагаемой методики не рассматривается ставшая хрестоматийной методика упреждающей диагностики сети. Не подвергая сомнению, важность упреждающей диагностики, следует заметить, что на практике она используется редко. Чаще всего (хотя это и неправильно) сеть анализируется только в периоды ее неудовлетворительной работы. В таких случаях локализовать и исправить имеющиеся дефекты сети требуется быстро. В данной работе основное внимание уделено диагностике канального уровня сети – поскольку это является первоочередной задачей при диагностике сети передачи данных.

Алгоритм поиска и устранения неисправности в общем виде состоит из восьми шагов (рис. 33):

1. Определение проблемы;
2. Сбор необходимой информации;
3. Оценка возможных сценариев решения проблемы и определение наиболее вероятных причин неисправности;
4. Разработка плана решения проблемы;
5. Осуществление действий в соответствии с составленным планом;
6. Оценка результатов;
7. Повторение последовательности шагов, в случае, если неисправность не была устранена;
8. Документирование изменений после успешного устранения неисправности.

Рисунок 33 – Последовательность устранения неполадок в сети


**Тестирование физического уровня.** Полноценно кабельная система может быть протестирована только специальным прибором - кабельным сканером. Другого способа не существует. Не имеет смысла заниматься трудоемкой процедурой выявления дефектов сети, если их можно локализовать одним нажатием клавиши на кабельном сканере. При этом прибор выполнит полный комплекс тестов на соответствие кабельной системы сети выбранному стандарту. Следует учитывать, что стандартное тестирование не позволяет проверить уровень шума создаваемого внешним источником в кабеле. Это может быть шум от люминесцентной лампы, силовой электропроводки, сотового телефона, мощного копировального аппарата и др. Для определения уровня шума кабельные сканеры имеют, как правило, специальную функцию. Поскольку кабельная система сети полностью проверяется только на этапе ее инсталляции, а шум в кабеле может возникать непредсказуемо, нет полной гарантии того, что шум проявится именно в период полномасштабной проверки сети на этапе ее инсталляции.

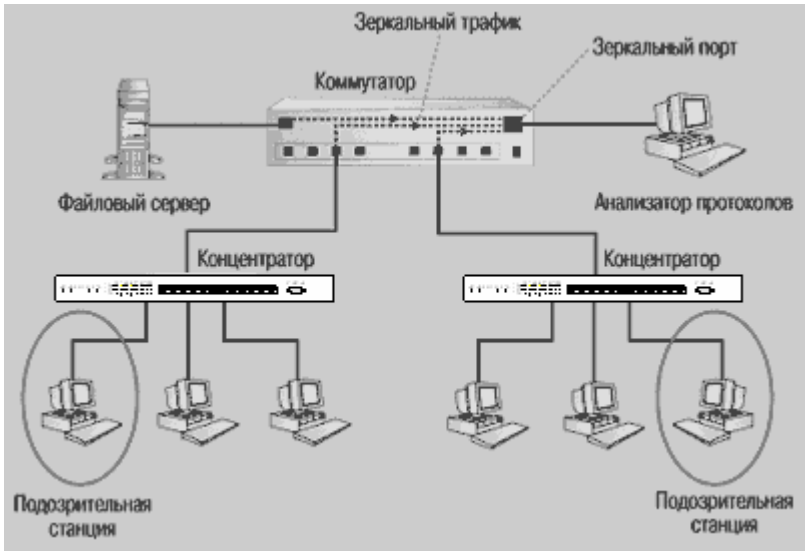
При проверке сети кабельным сканером вместо активного оборудования к кабелю подключаются с одного конца - сканер, с другого - инжектор. После проверки кабеля сканер и инжектор отключаются, и подключается активное оборудование: сетевые платы, концентраторы, коммутаторы.

**Тестирование канального уровня.** Любая методика тестирования сети существенно зависит от имеющихся в распоряжении системного администратора средств. В большинстве случаев необходимым и достаточным средством для обнаружения дефектов сети (кроме кабельного сканера) является анализатор сетевых протоколов.

Анализаторы могут быть подключены к коммутатору двумя способами.

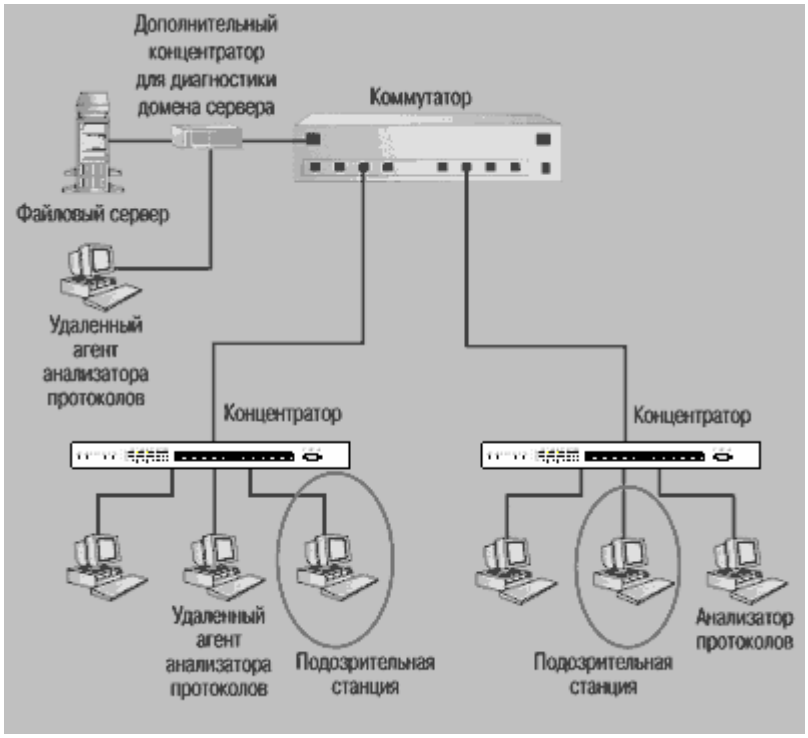
При первом способе (рис. 34) анализатор подключается к специальному порту (порту мониторинга или зеркальному порту) коммутатора, если таковой имеется, и на него по очереди направляется трафик со всех интересующих портов коммутатора.

Рисунок 34 - Подключение анализатора к порту мониторинга




Если в коммутаторе специальный порт отсутствует, то анализатор (или агент) следует подключать к портам интересующих доменов сети в максимальной близости к наиболее подозрительным станциям или серверу (рис. 35). Иногда это может потребовать использования дополнительного концентратора. Однако такой способ практически не используется в настоящее время, поскольку современное оборудование предоставляет подробную информацию и позволяет применять первый способ подключения анализатора.

Рисунок 35 - Подключение анализатора при отсутствии порта мониторинга




На рынке имеется множество разнообразных анализаторов протоколов - от чисто программных до программно-аппаратных. Несмотря на функциональную идентичность большинства анализаторов протоколов, каждый из них обладает теми или иными достоинствами и недостатками. В связи с этим следует обратить внимание на две важные функции, без которых эффективную диагностику сети провести будет затруднительно. Во-

первых, анализатор протоколов должен иметь встроенную функцию генерации трафика. Во-вторых, анализатор протоколов должен уметь отфильтровывать принимаемые кадры, то есть принимать не все кадры подряд, а, например, кадры определённого протокола. Если эта функция отсутствует, то при сильной загруженности сети, какой бы производительностью ни обладал компьютер, на котором установлен анализатор, последний будет терять кадры. Это особенно важно при диагностике быстрых сетей типа Fast Ethernet, FDDI и особенно Gigabit Ethernet.

**Измерение утилизации сети и установление корреляции между замедлением работы сети и перегрузкой канала связи.**

Утилизация канала связи сети - это процент времени, в течение которого канал связи передает сигналы, или иначе - доля пропускной способности канала связи, занимаемой кадрами, коллизиями и помехами. Параметр "Утилизация канала связи" характеризует величину загруженности сети.

Загруженность канала связи может влиять на время реакции прикладного программного обеспечения. И первоочередная задача состоит в определении наличия взаимозависимости между плохой работой прикладного программного обеспечения и утилизацией канала связи сети.

Многие источники упоминают о стандарте де-факто, в соответствии с которым для удовлетворительной работы сети Ethernet с общей шиной, утилизация канала связи "в тренде" (усредненное значение за 15 минут) не должна превышать 20%, а "в пике" (усредненное значение за 1 минуту) - 35-40%. Приведенные значения объясняются тем, что в сети Ethernet при утилизации канала связи, превышающей 40%, существенно возрастает число коллизий и, соответственно, время реакции прикладного ПО. Несмотря на то, что такие рассуждения в общем случае верны, безусловное следование подобным рекомендациям может привести к неправильному выводу о причинах медленной работы программ в сети.

Если в сети Ethernet в любой момент времени обмен данными происходит не более чем между двумя компьютерами, то любая сколь угодно высокая утилизация сети является допустимой.

Высокая утилизация канала связи сети только в том случае замедляет работу конкретного прикладного ПО, когда именно канал связи является "узким местом" для работы данного конкретного ПО.

Кроме канала связи узкие места в системе могут возникнуть из-за недостаточной производительности или неправильных параметров настройки сервера, низкой производительности рабочих станций, неэффективных алгоритмов работы самого прикладного ПО.

В какой мере канал связи ответственен за недостаточную производительность системы, можно выяснить следующим образом. Выбрав наиболее массовую операцию данного прикладного ПО (например, для банковского ПО такой операцией может быть ввод платежного поручения), следует определить, как утилизация канала связи влияет на время выполнения такой операции.

Проще всего это сделать, воспользовавшись функцией генерации трафика. С помощью этой функции интенсивность генерируемой нагрузки следует наращивать постепенно, и на ее фоне производить измерения времени выполнения операции. Фоновую нагрузку целесообразно увеличивать от 0 до 50-60% с шагом не более 10%.

Если время выполнения операции в широком интервале фоновых нагрузок не будет существенно изменяться, то узким местом системы является не канал связи. Если же время выполнения операции будет существенно меняться в зависимости от величины фоновой нагрузки (например, при 10% и 20% утилизации канала связи время выполнения операции будет значительно различаться), то именно канал связи, скорее всего, ответственен за

низкую производительность системы, и величина его загруженности критична для времени реакции прикладного ПО. Зная желаемое время реакции ПО, легко можно определить, какой утилизации канала связи соответствует желаемое время реакции прикладного ПО.

В данном эксперименте фоновую нагрузку не следует задавать более 60%. Даже если канал связи не является узким местом, при таких нагрузках время выполнения операций может возрасти вследствие уменьшения эффективной пропускной способности сети.

**Измерение числа коллизий в сети.**

Такой показатель как число коллизий становится неактуальным в настоящее время, поскольку практически повсеместно осуществлён переход на коммутируемую полнодуплексную среду передачи данных.

Если две станции коллизионного домена сети одновременно ведут передачу данных, то в домене возникает коллизия. Коллизии бывают трех типов: местные, удаленные, поздние.

Местная коллизия (local collision) - это коллизия, фиксируемая в коллизионном домене, где подключено измерительное устройство, в пределах передачи преамбулы или первых 64 байт кадра, когда источник передачи находится в домене.

В сетях 100BaseT (а также 10BaseT) станция определяет, что произошла локальная коллизия, если во время передачи кадра она обнаруживает активность на приемной паре (Rx).

Удаленная коллизия (remote collision) - это коллизия, которая возникает в другом физическом сегменте сети. Станция, работающая в полудуплексном режиме, узнает, что произошла удаленная коллизия, если она получает неправильно оформленный короткий кадр с неверной контрольной последовательностью CRC, и при этом отсутствует одновременная активность на приемной и передающей парах (Tx и Rx).

Поздняя коллизия (late collision) - это местная коллизия, которая фиксируется уже после того, как станция передала в канал связи первые 64 байт кадра. В сетях 10BaseT/100BaseT поздние коллизии часто фиксируются измерительными устройствами как ошибки CRC.

Даже если канал связи не является узким местом системы, коллизии несущественно, но замедляют работу прикладного ПО. Причем основное замедление вызывается не столько самим фактом необходимости повторной передачи кадра, сколько тем, что каждый компьютер сети после возникновения коллизии должен выполнять алгоритм отката (backoff algorithm): до следующей попытки выхода в канал связи ему придется ждать случайный промежуток времени, пропорциональный числу предыдущих неудачных попыток.

Следует учитывать, что не все измерительные приборы правильно определяют общее число коллизий в сети.

Практически все чисто программные анализаторы протоколов фиксируют наличие коллизии только в том случае, если они обнаруживают в сети фрагмент, то есть результат коллизии. При этом наиболее распространенный тип коллизий - происходящие в момент передачи преамбулы кадра (то есть до начального ограничителя кадра (SFD)) - программные измерительные средства не обнаруживают, так устроен набор микросхем сетевых плат Ethernet. Наиболее точно коллизии обнаруживают аппаратные измерительные приборы, например LANMeter компании Fluke.

Долю коллизий в общем числе кадров имеет смысл анализировать в момент активности подозрительных (медленно работающих) станций и только в случае, когда утилизация канала связи превышает 30%.

Коллизии в сети могут быть следствием перегруженности входных буферов коммутатора. Следует помнить, что коммутаторы при перегруженности входных буферов эмулируют коллизии, искусственно понижая скорость передачи рабочих станций сети. Этот механизм называется "управление потоком" (flow control).

**Измерение числа ошибок на канальном уровне сети.**

В сетях Ethernet наиболее распространенными являются следующие типы ошибок.

- Короткий кадр - кадр длиной менее 64 байт (после 8-байтной преамбулы) с правильной контрольной последовательностью. Наиболее вероятная причина появления коротких кадров - неисправная сетевая плата или неправильно сконфигурированный или испорченный сетевой драйвер;
- Ошибки контрольной последовательности (CRC error) - правильно оформленный кадр, но с неверной контрольной последовательностью (ошибка в поле CRC);
- Ошибка выравнивания (alignment error) - кадр, содержащий число бит, не кратное числу байт.
- Блики (ghosts) - последовательность сигналов, отличных по формату от кадров Ethernet, не содержащая разделителя (SFD) и длиной более 72 байт. Впервые данный термин был введен компанией Fluke с целью дифференциации различий между удаленными коллизиями и шумами в канале связи.

Блики являются наиболее коварной ошибкой, так как они не распознаются программными анализаторами протоколов по той же причине, что и коллизии на этапе передачи преамбулы. Выявить блики можно специальными приборами или с помощью метода стрессового тестирования сети.

Следует заметить, что степень влияния ошибок канального уровня сети на время реакции прикладного ПО сильно преувеличена.

В соответствии с общепринятым стандартом де-факто число ошибок канального уровня не должно превышать 1% от общего числа переданных по сети кадров. Как показывает опыт, эта величина перекрывается только при наличии явных дефектов кабельной системы сети.

Прежде чем анализировать ошибки в сети, следует выяснить, какие типы ошибок могут быть определены сетевой платой и драйвером платы на компьютере, где работает программный анализатор протоколов.

Работа любого анализатора протоколов основана на том, что сетевая плата и драйвер переводятся в режим приема всех кадров сети (promiscuous mode). В этом режиме сетевая плата принимает все проходящие по сети кадры, а не только широковещательные и адресованные непосредственно к ней, как в обычном режиме. Анализатор протоколов всю информацию о событиях в сети получает именно от драйвера сетевой платы, работающей в режиме приема всех кадров.

Не все сетевые платы и сетевые драйверы предоставляют анализатору протоколов идентичную и полную информацию об ошибках в сети. Сетевые платы 3Com вообще не выдают никакой информации об ошибках. Если установить анализатор протоколов на такую плату, то значения на всех счетчиках ошибок будут нулевыми.

EtherExpress Pro компании Intel сообщают только об ошибках CRC и выравнивания. Сетевые платы компании SMC предоставляют информацию только о коротких кадрах. NE2000 выдают почти полную информацию, выявляя ошибки CRC, короткие кадры, ошибки выравнивания, коллизии.

Сетевые карты D-Link и Kingstone сообщают полную, а при наличии специального драйвера - даже расширенную, информацию об ошибках и коллизиях в сети.

Ряд разработчиков анализаторов протоколов предлагают свои драйверы для наиболее популярных сетевых плат.

Для выявления ошибок на канальном уровне сети измерения необходимо проводить на фоне генерации анализатором протоколов собственного трафика. Генерация трафика позволяет обострить имеющиеся проблемы и создает условия для их проявления. Трафик должен иметь невысокую интенсивность (не более 100 кадров/с) и способствовать образованию коллизий в сети, то есть содержать короткие (<100 байт) кадры.

При выборе анализатора протоколов или другого диагностического средства внимание следует обратить, прежде всего, на то, чтобы выбранный инструмент имел встроенную функцию генерации трафика задаваемой интенсивности.

При первом проведении диагностике и наличии в ней проблем, не следует ожидать, что дефектен только один компонент.

Отсутствие ошибок на канальном уровне еще не гарантирует того, что информация сети не искажается. В начале данного раздела уже упоминалось, что влияние ошибок канального уровня на работу сети сильно преувеличено. Следствием ошибок нижнего уровня является повторная передача кадров. Благодаря высокой скорости сети Fast Ethernet и высокой производительности современных компьютеров, ошибки нижнего уровня не оказывает существенного влияния на время реакции прикладного ПО.

Таким образом, основная задача диагностики канального уровня сети - выявить наличие повышенного числа коллизий и ошибок в сети и найти взаимосвязь между числом ошибок и степенью загруженности канала связи. Все измерения следует проводить на фоне генерации анализатором протоколов собственного трафика.

Методика упреждающей диагностики заключается в следующем. Администратор сети должен непрерывно или в течение длительного времени наблюдать за работой сети. Такие наблюдения желательно проводить с момента ее установки. На основании этих наблюдений администратор должен определить, во-первых, как значения наблюдаемых параметров влияют на работу пользователей сети и, во-вторых, как они изменяются в течение длительного промежутка времени: рабочего дня, недели, месяца, квартала, года и так далее.

Наблюдаемыми параметрами обычно являются:

- параметры работы канала связи сети - утилизация канала связи, число принятых и переданных каждой станцией сети кадров, число ошибок в сети, число широковещательных и многоадресных кадров и так далее;
- параметры работы сервера - утилизация процессора сервера, число отложенных (ждущих) запросов к диску, общее число кэш-буферов, число "грязных" кэш-буферов и так далее.

Зная зависимость между временем реакции прикладного ПО и значениями наблюдаемых параметров, администратор сети должен определить максимальные значения параметров, допустимые для данной сети. Эти значения вводятся в виде порогов (thresholds) в диагностическое средство. Если в процессе эксплуатации сети значения наблюдаемых параметров превысят пороговые, то диагностическое средство проинформирует об этом событии администратора сети. Такая ситуация свидетельствует о наличии в сети проблемы.

Наблюдая достаточно долго за работой канала связи и сервера, можно установить тенденцию изменения значений различных параметров работы сети (утилизации ресурсов, числа ошибок и тому подобное). На основании таких наблюдений администратор может сделать выводы о необходимости замены активного оборудования или изменения архитектуры сети [22].

### 2.2. Архитектура Packet Sniffer SDK

Чтобы захватывать пакеты, идущие по сети, программа захвата должна взаимодействовать непосредственно с сетевым оборудованием. По этой причине ОС должна предоставлять набор действий по коммуникации и получению данных от сетевого адаптера. Задача этих примитивов – захват пакетов из сети и передача их запросившим программам. Здесь существует большая системная зависимость, поэтому реализация подобных действий различна на разных ОС. Захват пакетов должен быть быстр и рационален, поскольку эти действия, например, в высокоскоростной LAN с большой загруженностью траффика, должны минимизировать потери пакетов и использовать минимум системных ресурсов.

Одна из наиболее перспективних технологий захвата и анализа сетевого трафика представлена в продукте компании MicroOLAP (c) Packet Sniffer SDK. Packet Sniffer SDK

представляет собой библиотеку для захвата и анализа сетевого трафика для платформы Win32.

В соответствии с типовой архитектурой анализаторов сетевого трафика, на самом нижнем уровне иерархии лежит сетевой адаптер. Он используется для захвата пакетов, которые циркулируют в сети. Во время захвата сетевой адаптер обычно работает в особенном режиме, который заставляет адаптер принимать все пакеты (Promiscuous mode).

Драйвер захвата пакетов – низший программный уровень иерархии захвата. Эта часть работает в «режиме ядра» системы и взаимодействует с сетевым адаптером для получения пакетов. Он предоставляет программным приложениям набор функций для чтения и записи данных адаптера.

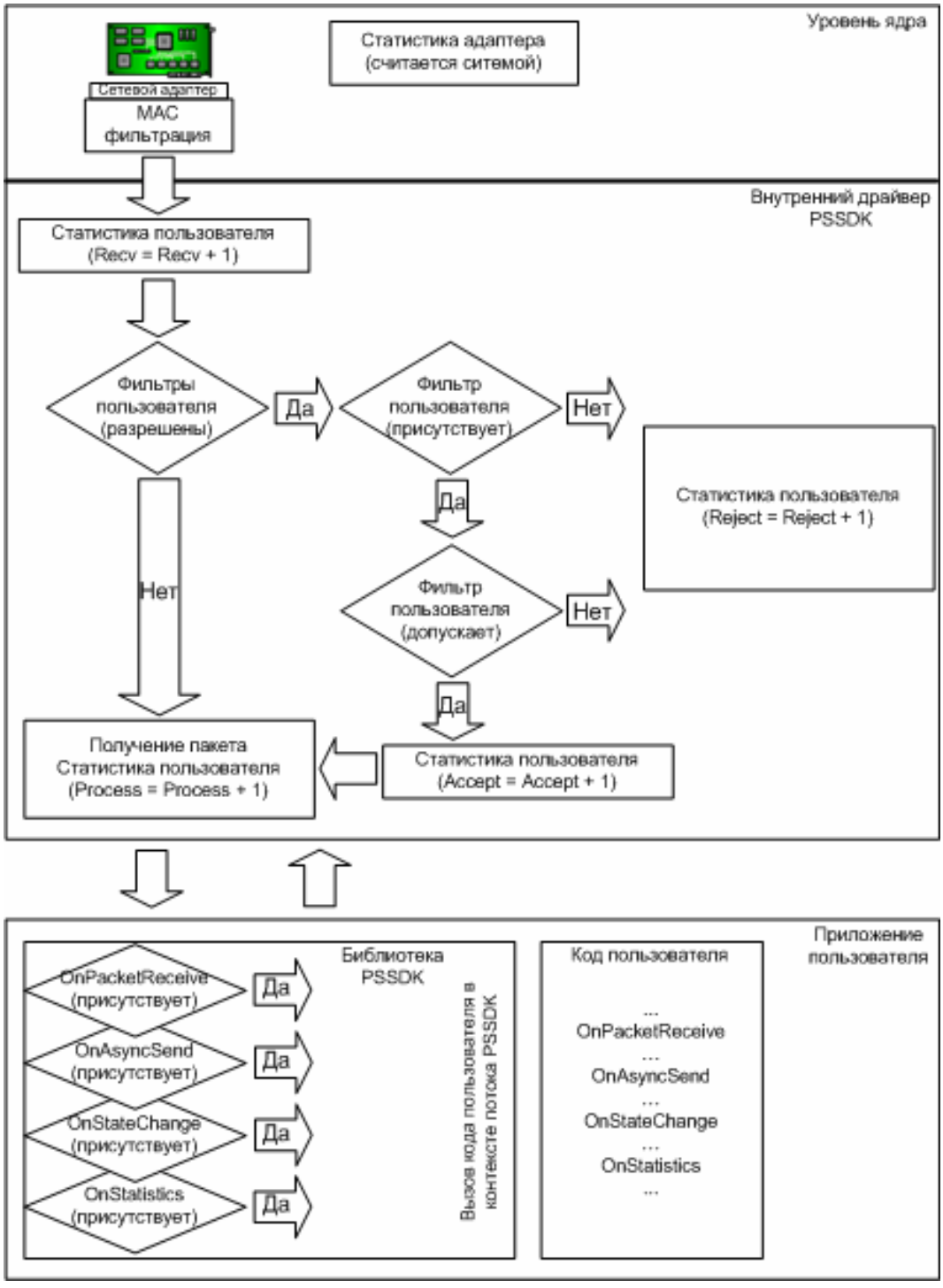
PSSDK (Packet Sniffer SDK) – работает в пользовательском режиме. Однако в отличие от стандартной архитектуры, данная библиотека содержит свой собственный динамически загружаемый драйвер, предоставляющий приложениям высокого уровня мощный интерфейс захвата. PSSDK представляет собой статически или динамически присоединяемую библиотеку, которая является частью приложения захвата пакетов.

Пользовательский интерфейс - наивысшая часть иерархии захвата, она управляет взаимодействие с пользователем и выдаёт результат захвата.

Программа пользовательского режима получает пакеты от системы, интерпертирует их и обрабатывает, выдавая пользователю в понятном виде.

На рис. 36 представлена диаграмма взаимодействия компонентов Packet Sniffer SDK в процессе захвата пакетов и передаче их приложению пользователя с ведением статистики захвата.

Рисунок 36 - Архитектура Packet Sniffer SDK




Основными особенностями Packet Sniffer SDK являются:

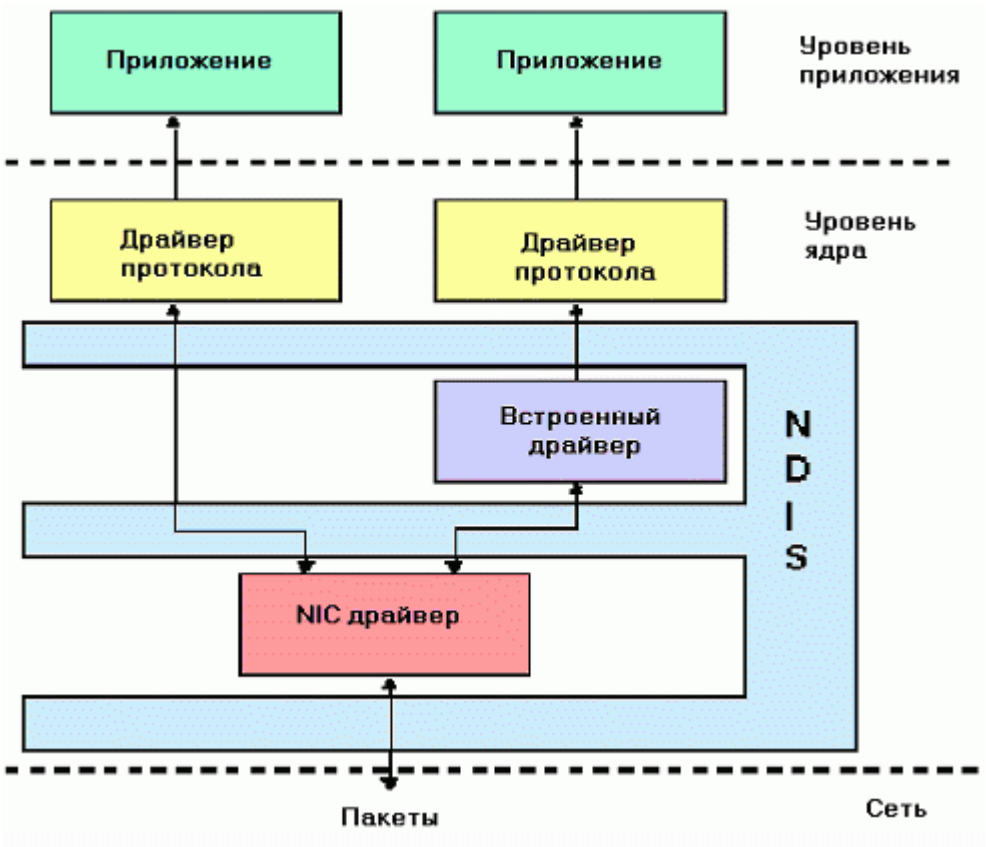
- Полноценная поддержка сетей 1GBit;
- Наличие внутреннего динамически загружаемого драйвера пакетов, что обуславливает отсутствие необходимости в предустановленных драйверах;
- Поддержка мультипроцессорных (SMP) систем;
- Поддержка новой технологии фильтрации пакетов – FastBPF, обеспечивающей в среднем в 6 раз большую производительность по сравнению с обычными BPF фильтрами;
- Поддержка BPF ассемблера для написания BPF/FastBPF фильтров.

**Взаимодействие с NDIS**

NDIS – это набор функций, которые устанавливают взаимодействие драйверов сетевых карт и драйверов протоколов (IP, IPX и так далее). Главное предназначение NDIS – действовать в качестве упаковщика, который позволяет драйверам протоколов посылать и получать пакеты из сети(WAN, LAN) вне зависимости от конкретного адаптера или данной Win32 ОС.

NDIS поддерживает 3 типа сетевых драйверов (рис. 37).

Рисунок 37 - NDIS структура с двумя схемами захвата




Типы сетевых драйверов:

1. Драйвер карты сетевого интерфейса (NIC). NIC драйверы взаимодействуют исключительно с физическим оборудованием на их нижнем уровне, а на верхнем присутствует интерфейс, позволяющий верхним драйверам посылать пакеты в сеть, использовать системные прерывания, сбрасывать текущее состояние NIC, останавливать NIC. NIC – драйвер не может общаться с приложениями пользовательского режима, только с промежуточными NDIS драйверами и драйверами протоколов;
2. Промежуточный драйвер общается с одной стороны с драйверами верхнего уровня, такими как драйвера транспорта, с другой с минипортами. Для минипорта промежуточный драйвер выглядит как драйвер протокола. Промежуточный драйвер может общаться с приложением, но только вместе с другими NDIS драйверами;
3. Транспортный драйвер или драйвер протокола. Драйвер протокола составляет сетевой стек протоколов, таких как IPX/SPX, TCP/IP. Транспортный драйвер обслуживает программы пользовательского режима, на своём верхнем уровне и соединяется с одним и более NIC драйвером или промежуточным NDIS-драйвером на нижнем уровне.

Драйвер захвата пакетов должен быть связан с сетевым драйвером, чтобы получать данные из сети и с пользовательским приложением для того, чтобы предоставлять ему пакеты данных. Таким образом, он внедрён в NDIS структуру как драйвер протокола. Это позволяет ему быть независимым от сетевой аппаратной архитектуры, но, тем не менее, работать под всеми интерфейсами, поддерживаемыми Windows. Однако, следует учитывать, что данный драйвер работает только на Ethernet адаптерах и в некоторых WAN соединениях в связи с ограничениями, налагаемыми архитектурой драйверов и фильтров. Также, надо учитывать, что WAN соединение понимается драйвером протокола как Ethernet NIC и каждый полученный пакет имеет искусственный Ethernet заголовок, созданный NDIS. Это позволяет драйверам протоколов работать на WAN соединениях без разницы, но предполагает также, что специфические пакеты, такие как PPP NCP-LCP не видны для драйвера протокола, поскольку PPP – соединение – виртуально. Это означает, что драйвер не сможет перехватывать пакеты данных такого типа.

**Процесс фильтрации пакетов**

Для каждого подключения между адаптером и программой сбора данных, драйвер создает фильтр и буфер. Один сетевой интерфейс может использоваться более чем одним приложением одновременно. Например, пользователь, который хочет фиксировать IP и UDP трафик в сети и сохранять их в двух отдельных файлах, может запустить два сеанса Sniffer на одном адаптере (но с различными фильтрами) одновременно. Первый сеанс поставит фильтр для пакетов IP (и буфер будет хранить их), а второй фильтровать UDP пакеты. В Windows NT это также возможно для одного приложения, чтобы получить пакеты от более чем одного интерфейса, открывая сеанс на каждом сетевом адаптере.

Механизм фильтрации, который присутствует в драйвере захвата пакетов, наследован из фильтра драйвера BPF, используемого в UNIX.

**BPF фильтр**

При использовании сетевого анализатора с выводом информации на консоль, одна из проблем, с которой сталкивается пользователь, заключается в том, что анализатор не будет успевать отображать все данные, поступившие из сети, и часть пакетов будет потеряна.

Это особенно актуально, если трафик очень плотный. Вторая проблема – пользователя могут интересовать только пакеты, адресованные выделенным хостам, а не все подряд. Решение заключается в фильтрации входящих сетевых пакетов по какому-либо определенному признаку, например по адресной части. Одним из вариантов решения является использование оператора условия if в коде сетевого анализатора, однако данное решение неэффективно. В этом случае приходится вытаскивать полный пакет из сети, на что отнимается процессорное время, затем анализатор вынужден «экзаменовать» заголовок каждого пакета перед принятием решения – отображать данные или нет. Оптимальным является решение отсеять лишние пакеты как можно раньше, на уровне драйвера сетевой карты. Это осуществляется при помощи пакетного фильтра.

Пакетный фильтр представляет собой последовательность инструкций, составленных в кодах псевдо-машинного языка, который называется BPF – Berkeley Packet Filter. Этот язык был разработан Стивом Маккеном (Steve McCanne) и Ван Якобсоном (Van Jacobson). BPF похож на язык ассемблер. В нем, как и в ассемблере, есть регистры, инструкции для загрузки и хранения операндов, выполнения арифметико-логических операций, условных и безусловных переходов. Для работы с операндами в BPF используются регистр-аккумулятор (или просто аккумулятор), индексный регистр, ячейка памяти и внутренний программный счетчик. Формат инструкции языка BPF определяет следующая структура:

```
struct sock_filter
{
   _u16 code;
   _u8 jt;
   _u8 jf;
   _u32 k;
}
```

Назначение полей структуры следующее:

- поле k – числовое значение операнда, с которым работает инструкция;
- поля jt (jump true) и jf (jump false) – меняют порядок выполнения инструкций;
- поле code – код выполняемой инструкции.

Существует 8 типов инструкций: BPF_LD, BPF_LDX, BPF_ST, BPF_STX, BPF_ALU, BPF_JMP, BPF_RET, BPF_MISC.

**BPF_LD.** Инструкция BPF_LD служит для загрузки в аккумулятор следующих величин:

константы (BPF_IMM);

блока данных, расположенных по фиксированному смещению (BPF_ABS);

блока данных, расположенных по смещению, которое является переменной величиной (BPF_IND);

длины блока данных (BPF_LEN);

значения, находящегося в ячейке памяти (BPF_MEM).

**BPF_ST.** Инструкция BPF_LDX служит для загрузки в индексный регистр следующих величин:

- константы (BPF_IMM);
- значения, находящегося в ячейке памяти (BPF_MEM);
- длины блока данных (BPF_LEN).

**BPF_ST.** Инструкция BPF_ST служит для загрузки аккумулятора в ячейку памяти.

**BPF_STX.** Инструкция BPF_STX служит для загрузки индексного регистра в ячейку памяти.

**BPF_ALU.** Инструкция BPF_ALU выполняет арифметико-логические между аккумулятором и индексным регистром или константой. Результат сохраняется в аккумуляторе.

**BPF_JMP.** Инструкция BPF_JMP изменяет порядок выполнения программы фильтрации. Данная инструкция может осуществлять как условный, так и безусловный переход между инструкциями. При безусловном переходе (BPF_JA, jump always) смещение задается 32-битным значением, при условном – 8-битным.

**BPF_MISC.** Инструкция BPF_MISC служит для копирования значения индексного регистра в аккумулятор и наоборот.

**BPF_RET.** Программа фильтрации выполняется для каждого пакета, поступающего на сетевой интерфейс. Результатом работы фильтра является целое положительное число, показывающее, сколько байт в принятом пакете будет доступно для дальнейшей обработки приложению пользователя. Если принятый пакет не удовлетворяет условиям фильтрации, он отбрасывается и программой фильтрации возвращается нулевое значение. Инструкция BPF_RET завершает выполнение программы фильтрации и возвращает число байт в пакете, доступных для дальнейшей обработки.

**Фильтрация в драйвере захвата пакетов**

Приложение, которому нужно установить фильтр на поступающие пакеты, может создать стандартный BPF фильтр и направить его драйверу, тогда процесс фильтрации будет исполняться в «режиме ядра». Важным моментом является то, что драйверу необходимо иметь возможность проверять код фильтра, направленного приложением. Фактически, BPF псевдо-машины могут исполнять арифметические операции. Поэтому деление на ноль, обращение к памяти по «невалидному» указателю неминуемо приведёт к краху системы. Так как это может быть использовано злоумышленником для того, чтобы привести систему к краху, то каждый фильтр, поступающий от приложения, проверяется прежде чем быть принятым. Если фильтр принят, он исполняет каждый раз, когда поступает новый пакет данных, отбрасывая те, которые не удовлетворяют условиям фильтра. Если пакет удовлетворяет этим условиям, он отправляется приложению или буферизуется в отдельной очереди, если приложение не готово принимать пакеты. Если никакого фильтра не установлено, то драйвер обрабатывает все пакеты. Фильтр устанавливается для пакета, когда тот ещё находится в памяти NIC драйвера без копирования его в драйвер захвата пакетов. Наиболее интересная черта, наследуемая драйвером захвата пакетов от BPF фильтра – возможность фильтровать не только пакеты, но и сами данные пакета.

## 2.3. Описание системы анализа трафика Sniffer

Система анализа трафика Sniffer представляет собой приложение, разработанное на базе Packet Sniffer SDK компании MicroOLAP (c), которое позволяет захватывать трафик высокоскоростных сетей Ethernet.

Packet Sniffer SDK выпускается в нескольких вариантах:

1. Статически присоединяемая библиотека (LIB edition);
2. Динамически присоединяемая библиотека (DLL edition);
3. Библиотека графических компонентов (VCL edition).

Приложение реализовано на языке программирования C++, на основе Packet Sniffer SDK VCL edition в среде Borland C++Builder 6. По сравнению с аналогичными продуктами, предназначенными для мониторинга и анализа сетей, разработанное приложение имеет значительно большую производительность, что позволяет избежать потери пакетов при высокой нагрузке сетей типа 1 Gigabit Ethernet. Увеличение производительности достигается за счёт возможности работы приложения на многопроцессорных системах (SMP), а также использования FastBPF фильтров.

Поддержка технологии фильтрации пакетов FastBPF включает оптимизированный компилятор BPF, преобразующий программу на ассемблере BPF в оптимизированные наборы инструкций процессора. В среднем фильтры FastBPF имеют в 6 раз большую производительность по сравнению с обычными фильтрами BPF.

Данное приложение представляет собой активный пакетный снифер, то есть помимо захвата трафика локальной сети позволяет генерировать трафик заданной интенсивности с задаваемыми пользователем MAC-адресами получателя и отправителя. Эта возможность позволяет проводить более глубокий анализ сети, включая проверку реакции на стрессовую нагрузку. Возможен также более подробный анализ качества обслуживания и базовый анализ трафика RTP, что имеет решающее значение при построении и анализе мультисервисных сетей передачи данных. В рамках данной дипломной работы была реализована базовая функциональность для анализа производительности, предоставляемого качества обслуживания и безопасности сетей.

Поскольку существует достаточное количество различных продуктов предоставляющих возможность анализа сетевого трафика, при разработке приложения Sniffer были учтены достоинства и недостатки существующих приложений.

Основным недостатком подобных средств является небольшой набор средств анализа – на данный момент отсутствуют стандартные средства анализа RTP и RTCP пакетов. Подобная функциональность присутствует в узкоспециализированных приложениях анализа качества потокового голоса или видео передаваемого в сети, однако они индивидуальны для каждого производителя оборудования и малопригодны для анализа сети в целом, особенно безопасности. Таким образом, сетевому администратору для полноценной диагностики и анализа сети необходимо иметь целый набор программ, реализующих отдельные возможности. Помимо высокой стоимости подобного решения, большинство программ – это дорогие коммерческие продукты, возникает проблема совместимости и потери производительности приложений.

В отличие от существующих приложений Sniffer является некоммерческим продуктом и предоставляет возможность производительного и разностороннего анализа предоставляемого качества обслуживания и безопасности сети.

В состав приложения Sniffer входит исполняемый модуль, содержащий также динамически загружаемый драйвер захвата пакетов и набор компилированных BPF фильтров для основных протоколов (ARP, DNS, FTP, HTTP, ICMP, IMAP4, POP3, SMTP, SSH, SSL, Telnet, WHO, UDP, TCP).

Данный программный продукт предполагается расширить средствами анализа захваченных пакетов – в настоящее время разрабатывается модуль восстановления и анализа TCP сессий, соответствующая функциональность поддерживается Packet Sniffer SDK. Данная функциональность необходима для более детального анализа безопасности сети и проверки возможности перехвата и восстановления паролей и другой конфиденциальной информации, а также является полезной для сетевых администраторов при анализе активности пользователей сети.

Также предполагается создание модуля анализа RTP пакетов, обеспечивающего высокопроизводительный детальный анализ RTP и RTCP пакетов, захваченных приложением. Основной проблемой анализа трафика RTP и RTCP является гибкий характер данных протоколов и широкие возможности их расширения, что значительно затрудняет задачу детектирования и анализа. Протоколы поддерживают возможность введения дополнительных полей и инкапсуляции нескольких протокольных блоков данных в протокольный блок данных нижележащего (транспортного) уровня.

Ограничением текущей версии продукта является тридцатидневный период бесплатного использования, обусловленный маркетинговой политикой компании MicroOLAP (c), владельца прав на Packet Sniffer SDK.

# 3. Практическое построение мультисервисной сети передачи данных

## 3.1. Схема сети. Оборудование. Адресация. Настройки.

В соответствии с необходимостью составления исходных документов для обеспечения работ по построению корпоративной мультисервисной сети пресс-центра Российско-Германского саммита, первоочередной задачей было построение логической схемы сети передачи данных.

При построении сети имели место следующие исходные требования:

1. Обеспечение 30 рабочих мест журналистов пресс-центра стационарным компьютерным оборудованием с доступом к сети Internet;
2. Обеспечение возможности беспроводного (WiFi) подключения портативных компьютеров к локальной сети с возможностью доступа к сети Internet;
3. Обеспечение резерва проводных подключений для портативных компьютеров, не оборудованных соответствующими адаптерами беспроводных сетей;
4. Обеспечение проводной и беспроводной (WiFi) телефонной связи для технических сотрудников с сохранением их рабочих телефонных номеров;
5. Простота и удобства подключения новых хостов к сети;
6. Обеспечение высокой надёжности и производительности сети.

В соответствии с требуемой функциональностью было принято решение использовать следующее оборудование:

- маршрутизатор – Cisco 2811 Integrated Services Router
- коммутатор – Cisco 2960 Catalyst Switch
- точки приема – Cisco Aironet 1231 Access Point

**Cisco 2811 Router:**

- обеспечивает производительность различных услуг (таких как передача потокового голоса и обеспечение безопасности) на скорости носителя;
- высокая производительность системы;
- поддержка более 90 существующих и вновь создаваемых модулей;
- 2 интегрированных порта 10/100 Fast Ethernet;
- опциональная поддержка PoE (Power over Ethernet - питание по Ethernet);
- встроенное шифрование;
- поддержка SDM (Security Device Manager), обеспечивающего простоту управления;
- поддержка до 1500 VPN туннелей при использовании модуля AIM-EPII-PLUS;
- антивирусная защита с помощью NAC (Network Admission Control);
- функции обнаружения и предотвращения вторжения – система IPS (Intrusion Preventing System);
- функции программного межсетевого экрана (IOS Firewall);
- поддержка аналоговых и цифровых голосовых звонков;
- опциональная поддержка голосовой почты;
- опциональная поддержка Cisco CME (CallManager Express) для локальной обработки вызовов (до 36 IP-телефонов);
- опциональная поддержка SRST (Survivable Remote Site Telephony) для локальной поддержки голосовых вызовов (до 36 IP-телефонов).

**Cisco 2960 Catalyst Switch:**

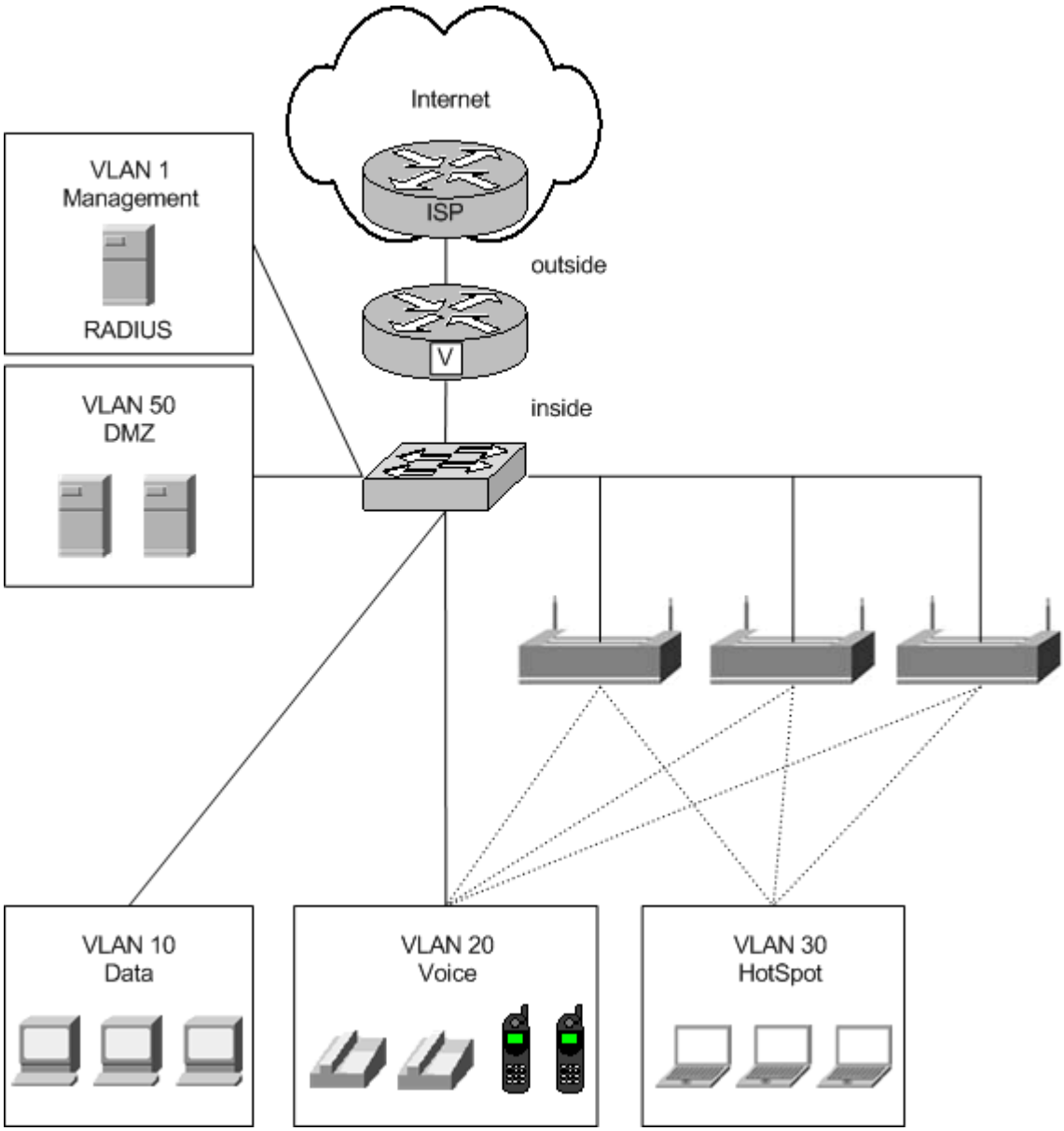
- интегрированная безопасность, включая NAC (Network Admission Control);
- поддержка QoS;
- 48 интегрированных портов 10/100 Fast Ethernet;

- 2 интегрированных порта Gigabit Ethernet.

**Cisco AiroNet 1231 Access Point:**
поддержка стандартов IEEE 802.11a/b/g;
поддержка питания по Ethernet;
поддержка средств управления;
интегрированные функции безопасности.

В соответствии с приведёнными требования была разработана сеть, схема которой представлена на рис. 38.

Рисунок 38 - Схема сети Российско-Германского саммита




Маршрутизатор выполняет функции маршрутизации трафика между VLAN-ами, отвечает за управление установлением соединения телефонных вызовов, является межсетевым экраном и обеспечивает контроль внутреннего и внешнего трафика.

Коммутатор осуществляет подключение точек приема, устройств уровня доступа (рабочие станции и IP-телефоны), организует виртуальные ЛВС, предоставляет функции QoS, обеспечивает безопасность на уровне портов.

Точки приема осуществляют подключение устройств уровня доступа (рабочие станции и IP-телефоны), предоставляет функции QoS и безопасности.

Схема адресации сети приведена в таблице 7.

Таблица 6. Схема адресации сети

| № | Название | Адрес | Описание |
|---|---|---|---|
| VLAN 1 | Management | 192.168.0.0/24 | Управляющий доступ к оборудованию производится только из Management VLAN, здесь же располагается RADIUS сервер. |
| VLAN 10 | Data | 192.168.10.0/24 | Сеть для стационарных рабочих станций по кабельному подключению |
| VLAN 20 | Voice | 192.168.20.0/24 | Сеть для голосового трафика (кабельные и радио IP-телефоны) |
| VLAN 30 | HotSpot | 192.168.30.0/24 | Сеть беспроводного доступа для журналистов с портативными компьютерами |
| VLAN 40 | Unused | | VLAN для неиспользуемых портов коммутатора (как составной компонент системы безопасности) |
| VLAN 50 | DMZ | 217.80.159.0/29 | Сеть для серверов публичного доступа (в частности, Proxy-сервер). |

В соответствии с данной схемой, на локальном маршрутизаторе устанавливается программное обеспечение, осуществляющее маршрутизацию звонков. Данная модель маршрутизатора позволяет устанавливать Cisco Call Manager Express, который поддерживает до 36 IP-телефонов.

Таким образом, схема представляет собой распределенную систему, обеспечивающую локальную обработку вызовов с обеспечением необходимых механизмов качества сервиса (QoS).

В такой схеме единственный сервер CallManager, управляет установлением телефонных соединений и функционированием телефонных аппаратов, расположенных в пределах локальной IP сети.

Основные характеристики предложенной модели построения сети IP телефонии:

- для подключения к телефонной сети общего пользования (ТФОП), подключения аналоговых телефонов и факсовых аппаратов и стыковки с существующими УАТС могут использоваться голосовые шлюзы;
- в пределах локальной сети возможно использование кодека G.711 (несжатый голос);
- для экономного использования полосы пропускания на каналах WAN может быть использовано сжатие голоса;
- Cisco CallManager контролирует использование полосы пропускания на каналах WAN между удаленными офисами и принимает решение о разрешении/запрете установления телефонного соединения на основе информации о наличии свободной полосы пропускания (call admission control);
- поддержка механизмов обеспечения качества сервиса (QoS) в пределах распределенной IP сети является критично важной для обеспечения качественной работы различных приложений (это особенно важно для голосовых приложений).

Подключение проводных IP телефонов к сети передачи данных осуществляется посредством одного кабеля (рис. 16), что обеспечивает простоту установки, отсутствие дополнительного расхода портов коммутатора и необходимости изменение кабельной инфраструктуры. При этом предполагается разнесение рабочих станций и телефонных аппаратов в различные виртуальные локальные сети, что позволит обеспечивать требуемое качество обслуживания и увеличит безопасность.

Для подключения проводных IP телефонов к электрической сети предполагается использовать технологию Inline Power, что обеспечит следующие преимущества: во-первых, не потребуется локальная розетка электропитания для каждого телефонного аппарата, и, во-вторых, этот способ также позволяет централизовать средства управления и обеспечения надёжности электропитания.

Поскольку требуется обеспечение надёжной работы телефонной сети при сбоях электропитания (до 4 часов работы в автономном режиме), используются источники бесперебойного питания (UPS); при этом UPS могут использоваться только для коммутаторов, поддерживающих технологию Inline Power и других важных сетевых устройств и серверов (в том числе сервера Call-Manager).

После установки и подключения оборудования, потребовалось провести его настройку, в соответствии с требования к сети. Помимо базовых настроек первоочередное значение имели настройки безопасности [27].

В рамках обеспечения безопасности были отключены все неиспользуемые службы. В том числе различные TCP и UDP сервисы, служба finger (по которой можно получить некоторую конфиденциальную информацию), отключен протокол bootp, отключен snmp, так как его использование не предполагалось:

```
no service finger
no service pad
no service tcp-small-servers
no service udp-small-servers
no snmp-server
no ip bootp server
```

Также была отключена маршрутизация по адресу источника - no ip source-route.

Ряд служб следовало включить:

- Шифрование паролей;
- Входящие и исходящие TCP keepalive сообщения;
- Службу временных отметок (необходимо для датирования лог-информации);
- Службу простановки sequence number в лог сообщениях;
- CEF (Cisco express forwarding – необходимо для работы RPF – reverse pass forwarding).

```
service password encryption
service tcp-keepalives-in
service tcp-keepalives-out
service timestamps debuf datetime localtime show-timezone msec
service timestamps log datetime localtime show-timezone msec
service sequence-numbers
ip cef
```

Были настроены информационные баннеры, появляющиеся при входе на устройство:

```
banner # <message> #
banner motd # <message> #
banner login # <message> #
```

В качестве сообщения, выводимого в баннере, было использовано одно из рекомендуемых: “Authorized access only! This system is the property of <company name>. Disconnect IMMEDIATELY as you are not an authorized user! Contact <administrator email address> <administrator phone number>”.

Так же были наложены ограничения на возможные пароли и включен учет ошибочных попыток аутентификации:

```
security passwords min-length 8
```

security authentication failure rate 3 log

Заданы параметры сбора лог-информации:
logging on
logging 192.168.5.100 ! log-server
logging console critical
logging trap debugging
logging buffered 32000

На интерфейсной базе:

- Отключены направленные широковещания.
- Отключен proxy-arp.
- Отключены перенаправления.
- Включена опция обратной проверки (RPF – reverse pass forwarding).

no ip directed-broadcast
no ip proxy-arp
no ip redirects
ip verify unicast reverse-path

Использования RPF является крайне важным моментом, так как это частично помогает бороться со спуффингом IP адресов. Суть данной технологии заключается в том, что маршрутизатор проверяет, на правильном ли интерфейсе получен пакет с данным адресом источника. Если согласно маршрутизирующей информации сеть источника находиться за другим интерфейсом, пакет отбрасывается. Проверка происходить не по таблицам маршрутизации, так как это требует значительных временных затрат, а по специальной базе данных, созданной CEF.

Также требовалось настроить IP телефонию. Поскольку телефоны в данной сети должны были получать IP адреса автоматически, первоочередной задачей было создание адресного пула для IP телефонов (данный пул создаётся для VLAN Voice отдельно от адресных пулов для клиентских рабочих станций – VLAN Data, HotSpot):

ip dhcp excluded-address 192.168.20.0 192.168.3.10
ip dhcp excluded-address 192.168.20.250 192.168.3.254
ip dhcp pool IpPhones
network 192.168.20.0 255.255.255.0
option 150 ip 192.168.20.254
default-router 192.168.20.254

Данный пул DHCP, помимо информации об адресе, выделенном клиенту, предоставляет клиентскому телефону информацию об адресе tftp-сервера, содержащего конфигурационные файлы для телефонов и файлы firmware для проводных IP телефонов.

Для доступа к телефонам за пределами локальной сети были определены правила трансляции номеров:

voice translation-rule 10
 rule 1 /^3822\(.....\)/ /\1/
 rule 2 /\(.....\)/ /8\1/
voice translation-rule 20
 rule 1 /^3822555555/ /103/
 rule 2 /^3822777777/ /105/
voice translation-profile Filter_3822
 translate calling 10
 translate called 20

translate redirect-called 20

**Настройка сервисов IP-телефонии**

Включение IP телефонии - telephony-service.
Определение firmware файлов для различных моделей телефонов:
load 7960-7940 P00307010200
load 7914 S00104000100

Указание максимального числа телефонов и максимального числа номеров директорий:
max-ephones 30
max-dn 150

Задание адреса интерфейса и порта, использующегося Cisco Call Manager-ом:
ip source-address 192.168.20.254 port 2000

Определение форматы даты и системного сообщения:
time-format 24
date-format dd-mm-yy
system message Elecs.Com Ltd.

Создание конфигурационных файлов - create cnf-files version-stamp 7960 Apr 21 2006 16:54:19.

Задание максимального числа одновременно идущих конференций - max-conferences 4.

Определение музыкальной заставки проигрываемой при удержании звонка - moh music-on-hold.au.

Кроме этого достаточно было задать номера директорий для телефонов (ephone-dn) и определить сами IP телефоны (ephone):

ephone-dn 1 dual-line
number 101
label XXX
description XXX-phone
transfer-mode consult

ephone 1
username "xxx"
mac-address 000F.8FFB.A548
type 7960
button 1:11 2m1

### 3.2. Меры обеспечения требуемого качества обслуживания

После того как сеть спроектирована, необходимо оценить предполагаемую загруженность сети и используемые в ней протоколы с целью определения необходимых средств для обеспечения требуемого качества обслуживания. В случае если обеспечить требуемое качество обслуживания требуется в уже существующей сети, необходимо определить состояние сети и используемые в ней протоколы. Для этих целей используется анализатор сетевых протоколов.

Для обеспечения требуемой пропускной способности применяются следующие техники:

- Использование очередей: основывается на передаче пакетов через конкретный интерфейс в соответствии с заданными приоритетами, позволяет обрабатывать

интенсивные потоки, управлять нагрузкой сети, приоритизировать трафик, резервировать пропускную способность;

- Сжатие заголовков: в IP сетях голос передаётся при помощи протокола реального времени Real-Time Transport Protocol (RTP), который переносится протоколом UDP, датаграммы UDP инкапсулируются в пакеты IP. Таким образом, составной заголовок RTP/UDP/IP достигает 40 байт. Это достаточно большая величина, поскольку объем данных, предаваемых в одном пакете, в большинстве случаев составляет 20 байт. Применение сжатия заголовков (CRTP) уменьшает размер заголовка до 2-4 байт.
- Контроль установления вызова: данный механизм расширяет возможности обеспечения качества обслуживания, обеспечивая защиту голосового трафика от негативного влияния другого голосового трафика путём ограничения количества одновременно установленных вызовов.
- Фрагментация и чередование: при фрагментации большие пакеты разбиваются на более мелкие, между которыми передаются голосовые пакеты, что позволяет избежать задержек, связанных с выводом больших пакетов в интерфейс.

В основе обеспечения качества обслуживания лежит возможность сетевых устройств распознавать и группировать специфические пакеты. Процесс распознавания получил название “классификация пакетов”. После классификации пакет должен быть помечен соответствующим образом, для чего выставляются соответствующие флаги в IP заголовке.

Классификация пакетов – достаточно ресурсоёмкий процесс, поэтому классификация должна происходить как можно ближе к краю сети. В ядре классификация должна быть максимально упрощена, это достигается за счёт маркирования пакетов - установки байта типа сервиса (Type of Service) в заголовке IP.

Три старших бита байта (ToS) называются битами старшинства IP (IP Precedence). В настоящее время большинство приложений и производителей оборудования поддерживают установку и распознавание битов старшинства IP. Часто для определения дифференцированных классов сервиса (Differentiated Services classes) используются шесть старших битов, называемых Differentiated Services Code Point.

В данной сети классификация пакетов происходила на точках доступа беспроводной сети и коммутаторе, к которому осуществлялось подключение клиентских станций.

В качестве основания классификации использовались значения Differentiated Services Code Point входящих пакетов.

При включении QoS коммутатор производит автоматическую трансляцию метки CoS (Class of Service) в метку DSCP. Поскольку требовалось выставлять метку CoS на основании DSCP, модификация DSCP коммутатором была запрещена: no mls qos rewrite ip dscp.

Было определено два класса трафика – для RTP и RTCP трафика. Поскольку пакетов RTP и RTCP требуют более приоритетное обслуживание, соответствующее программное обеспечение выставляет для данных пакетов метку DSCP в одно из принятых значений.

```
class-map match-all QoS-VoIP-RTP
  match ip dscp ef
class-map match-all QoS-VoIP-Control
  match ip dscp cs3  af31
```

В соответствии с заданными классами была определена policy-map, задающая пороговое значение загрузки порта и действие в случае её достижения (в данном случае, передачу только приоритетных пакетов):

```
policy-map QoS-Phone
  class QoS-VoIP-RTP
    police 1000000 8000 exceed-action policed-dscp-transmit
  class QoS-VoIP-Control
    police 1000000 8000 exceed-action policed-dscp-transmit
```

После создания policy-map была применена к интерфейсам, находящимся в VLAN 20 – то есть интерфейсам подключения проводных IP телефонов:

```
interface FastEthernet0/xx
 switchport access vlan 20
 switchport mode access
 switchport port-security
 switchport port-security maximum 5
 switchport port-security aging time 1
 switchport port-security violation protect
 switchport port-security aging type inactivity
 switchport port-security aging static
 srr-queue bandwidth share 10 10 60 20
 srr-queue bandwidth shape  10  0  0  0
 service-policy input QoS-Phone
 spanning-tree portfast
 spanning-tree bpdufilter enable
```

На транковых соединениях (соединения с точками доступа и маршрутизатором) – предполагается, что классификация пакетов уже выполнена и выставлено соответствующее значение CoS: mls qos trust cos.

Помимо этого осуществляется конфигурирование очередей и пороговых значений для них, а также распределение пакетов разных приоритетов по очередям.

```
mls qos srr-queue input bandwidth 90 10
```

Устанавливает веса для входящих очередей, в соответствии с которыми они обслуживаются алгоритмом shared round robin (SRR). SRR использует для приоритетной очереди процент пропускной способности заданный командой mls qos srr-queue input priority-queue (по умолчанию 50), оставшаяся пропускная способность разделяется между обеими очередями в соответствии с установленными весами.

```
mls qos srr-queue input threshold 1 8 16
mls qos srr-queue input threshold 2 34 66
```

Устанавливает по два пороговых значения для каждой из входящих очередей, пороговое значение – процентное соотношение выделенных для очереди буферов к общему количеству буферов.

```
mls qos srr-queue input buffers 67 33
```

Устанавливает распределение буферов (в процентном соотношении) между очередями.

```
mls qos srr-queue input cos-map queue 1 threshold 2  1
mls qos srr-queue input cos-map queue 1 threshold 3  0
mls qos srr-queue input cos-map queue 2 threshold 1  2
mls qos srr-queue input cos-map queue 2 threshold 2  4 6 7
mls qos srr-queue input cos-map queue 2 threshold 3  3 5
```

Задаёт соответствие значений CoS номерам входящих очередей и пороговых значений.

```
mls qos srr-queue input dscp-map queue 1 threshold 2  9 10 11 12 13 14 15
mls qos srr-queue input dscp-map queue 1 threshold 3  0 1 2 3 4 5 6 7
mls qos srr-queue input dscp-map queue 1 threshold 3  32
mls qos srr-queue input dscp-map queue 2 threshold 1  16 17 18 19 20 21 22 23
mls qos srr-queue input dscp-map queue 2 threshold 2  33 34 35 36 37 38 39 48
```

```
mls qos srr-queue input dscp-map queue 2 threshold 2  49 50 51 52 53 54 55 56
mls qos srr-queue input dscp-map queue 2 threshold 2  57 58 59 60 61 62 63
mls qos srr-queue input dscp-map queue 2 threshold 3  24 25 26 27 28 29 30 31
mls qos srr-queue input dscp-map queue 2 threshold 3  40 41 42 43 44 45 46 47
```

Задаёт соответствие значений DSCP номерам входящих очередей и пороговых значений.

```
mls qos srr-queue output cos-map queue 1 threshold 3  5
mls qos srr-queue output cos-map queue 2 threshold 3  3 6 7
mls qos srr-queue output cos-map queue 3 threshold 3  2 4
mls qos srr-queue output cos-map queue 4 threshold 2  1
mls qos srr-queue output cos-map queue 4 threshold 3  0
```

Задаёт соответствие значений CoS номерам исходящих очередей и пороговых значений.

```
mls qos srr-queue output dscp-map queue 1 threshold 3  40 41 42 43 44 45 46 47
mls qos srr-queue output dscp-map queue 2 threshold 3  24 25 26 27 28 29 30 31
mls qos srr-queue output dscp-map queue 2 threshold 3  48 49 50 51 52 53 54 55
mls qos srr-queue output dscp-map queue 2 threshold 3  56 57 58 59 60 61 62 63
mls qos srr-queue output dscp-map queue 3 threshold 3  16 17 18 19 20 21 22 23
mls qos srr-queue output dscp-map queue 3 threshold 3  32 33 34 35 36 37 38 39
mls qos srr-queue output dscp-map queue 4 threshold 1  8
mls qos srr-queue output dscp-map queue 4 threshold 2  9 10 11 12 13 14 15
mls qos srr-queue output dscp-map queue 4 threshold 3  0 1 2 3 4 5 6 7
```

Задаёт соответствие значений DSCP номерам исходящих очередей и пороговых значений.

```
mls qos queue-set output 1 threshold 1 138 138 92 138
mls qos queue-set output 1 threshold 2 138 138 92 400
mls qos queue-set output 1 threshold 3 36 77 100 318
mls qos queue-set output 1 threshold 4 20 50 67 400
mls qos queue-set output 2 threshold 1 149 149 100 149
mls qos queue-set output 2 threshold 2 118 118 100 235
mls qos queue-set output 2 threshold 3 41 68 100 272
mls qos queue-set output 2 threshold 4 42 72 100 242
```

Устанавливает пороговые значения для WTD (Weighted Tail Drop), алгоритма обеспечивающего отбрасывание излишних пакетов, обеспечивая доступность буферов. Также задаёт максимальное количество выделенных буферов для исходящей очереди для каждого порта.

```
mls qos queue-set output 1 buffers 10 10 26 54
mls qos queue-set output 2 buffers 16 6 17 61
```

Распределяет количество буферов, доступное каждой исходящей очереди.

Конфигурация обеспечения требуемого качества обслуживания на точках доступа осуществляется подобным образом.

При конфигурировании качества обслуживания на маршрутизаторе были определены два класса (потоковые данные и управляющий трафик), классифицирующие трафик на основе значений DSCP:

```
class-map match-any QoS-VoIP-RTP
 match ip dscp ef
class-map match-any QoS-VoIP-Control
```

```
 match ip dscp cs3
 match ip dscp af31
```

В соответствии с заданными классами была определена policy-map, задающая пороговое значение загрузки порта и действие в случае её достижения (в данном случае, передачу приоритетных пакетов):

```
policy-map QoS-Phone
  class QoS-VoIP-RTP
  set cos dscp
  priority percent 70
  class QoS-VoIP-Control
  bandwidth percent 5
  set cos dscp
  class class-default
  set cos 0
  fair-queue
```

Данная policy-map использует карту трансляции значений DSCP в CoS - set cos dscp. Поскольку для всего неклассифицированного трафика значение CoS выставляется равным 0, карте трансляции достаточно будет иметь две записи:

```
mls qos map dscp-cos ef to 5
mls qos map dscp-cos cs3 af31 to 3
```

После создания policy-map была применена на оба физических интерфейса:

```
interface FastEthernet0/x
 description default_gateway
 no ip address
 duplex auto
 speed auto
 service-policy output QoS-Phone
```

Таким образом, на физических интерфейсах были организованы LLQ (Low Latency Queuing) очереди. LLQ позволяет обеспечить приоритет определённому классу трафика, и гарантированную минимальную пропускную способность для других классов. При перегрузке сети приоритетный трафик удерживается в пределах заданного уровня, что позволяет избежать блокировки менее приоритетного трафика.

LLQ позволяет указать длину очереди, при превышении которой маршрутизатор отбрасывает поступающие пакеты. Также имеется класс “по умолчанию”, использующийся для обработки всего неклассифицированного остальными классами трафика. Этот класс в данном случае сконфигурирован на основе справедливой очереди (fair-queue), это означает, что каждый из неклассифицированных потоков получит примерно равную долю от оставшейся пропускной способности.

Трафик идущий через интерфейс сначала классифицируется с использованием class-map. Существует четыре класса трафика для LLQ: один приоритетный, два с гарантированной пропускной способностью и один класс “по умолчанию”. В данном случае используется один приоритетный класс (class QoS-VoIP-RTP), один класс с гарантированной пропускной способностью (class QoS-VoIP-Control) и класс по умолчанию (class class-default). Трафик приоритетного класса помещается в приоритетную очередь, классов с гарантированной полосой пропускания в очереди с зарезервированной полосой пропускания. Трафик класса “по умолчанию” использует очередь “по умолчанию”, где каждый из неклассифицированных потоков получит примерно равную долю от оставшейся полосы пропускания.

Планировщик обслуживает очереди таким образом, что сначала выводится приоритетный трафик, пока не будет достигнута установленная граница используемой данным трафиком пропускной способности, если данная пропускная способность востребована трафиком из зарезервированных очередей (то есть имеет место перегрузка сети). При переполнении приоритетной очереди в момент перегрузки сети, вновь поступающие приоритетные пакеты будут отбрасываться. Зарезервированные очереди обслуживаются в соответствии с запрошенной полосой пропускания, которую планировщик использует для вычисления веса. Вес определяет, как часто обслуживается очередь и сколько байт передается за одно обслуживание. Работа планировщика основана на алгоритме weighted fair queuing (WFQ).

Поскольку пропускная способность канала достаточно велика, решено было не использовать сжатие заголовков RTP и контроль установления вызова.

На данной сети в лабораторных условиях была опробована предложенная модель анализа и диагностики. Поскольку данная сеть имеет компактное ядро, которым является коммутатор Cisco 2960 Catalyst Switch, то в соответствии с предлагаемой методикой анализа, для захвата трафика используются зеркальные (span) порты. Анализатор снимает поток пакетов с зеркального порта и осуществляет его разбор.

Следует заранее оговорить некоторые ограничения. В случае пиковой нагрузки на серверы, их суммарный трафик может превзойти возможности порта, к которому подключен анализатор. В таком случае коммутатор производит отбрасывание не поместившихся кадров.

Схему подключения рабочей станции с функциями мониторинга и анализа см. на рис. 44.

Для практической реализации SPAN технологии необходимо:

1. Объявить источник сессия мониторинга;
2. Объявить назначение сессии мониторинга.

При необходимости анализа трафика целой виртуальной сети это производится при помощи команд:

```
Switch(config)# monitor session 1 source vlan xx both
Switch(config)# monitor session 1 destination interface fast 0/xx
```

При обнаружении подозрительного хоста в сети необходимо создать еще одну сессию, которая весь его трафик будет отправлять в зеркальный порт на анализатор:

```
Switch(config)# monitor session 2 source interface fast 0/xx both
Switch(config)# monitor session 2 destination interface fast 0/xx
```

# Заключение

В результате проделанной работы были проанализированы технологии, применяемые для построения мультисервисных сетей передачи данных. Изучены возможности различного сетевого оборудования по обеспечению телефонной, факсимильной и видео связи, и параметры его конфигурации. Подробно изучена архитектура Cisco Architecture for Voice Video and Integrated Data, инфраструктура мультисервисных сетей и сигнальные протоколы, применяемые в данных сетях. Проанализированы существующие методики диагностики и анализа локальных сетей передачи данных.

Создано приложение для анализа локальных сетей высокой производительности (до 1 Gbit/sec), представляющее собой активный сниффер пакетов с возможностью детального анализа захваченных пакетов. Приложение реализовано на основе Packet Sniffer SDK компании MicroOLAP и в отличие от подобных программ обладает большей производительностью и не требует предустановленного драйвера захвата пакетов.

На реальном оборудовании в ходе практической части работ была отработана базовая настройка и настройка основных механизмов обеспечения требуемого качества обслуживания и контроля установления вызова, опробована предложенная методика анализа локальных сетей передачи данных с использованием созданного приложения.

Также была предложена архитектура мультисервисной сети передачи данных пресс-центра Российско-Германского саммита на высшем уровне и подготовлена соответствующая документация. Проведены пуско-наладочные работы по инсталляции данной сети.

## Список литературы


1. Диагностика и анализ локальных сетей [Электронный ресурс]: — Электрон. дан. — КомпьютерМастер, 2004. — Режим доступа: http://www.computermaster.ru/articles/landiagnost.html., свободный.
2. Balenson D. Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes and Identifiers [Электронный ресурс]: — Электрон. дан. - RFC 1423, TIS, IAB IRTF PSRG, IETF PEM WG, 1993.
3. Braden R. Requirements for Internet Hosts - Application and Support [Электронный ресурс]: — Электрон. дан. - STD 3, RFC 1123, Internet Engineering Task Force, 1989.
4. Busse I., Deffner B., SchulzrinneH. Dynamic QoS control of multimedia applications based on RTP [Электронный ресурс]: — Электрон. дан. - Computer Communications, 1996.
5. Cadzow J. A. Foundations of digital signal processing and data analysis [Электронный ресурс]: — Электрон. дан. - N.-Y.: Macmillan, 1987.
6. Cisco Voice Over IP Version 4.2 [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2004. — 1 электрон. опт. диск (CD-ROM)
7. Clark D. D., Tennenhouse D. L. Architectural considerations for a new generation of protocols in SIGCOMM Symposium on Communications Architectures and Protocols // Computer Communications Review. — 1990. — № 20. — С. 200–208.
8. Crocker D. Standard for the Format of ARPA Internet Text Messages [Электронный ресурс]: — Электрон. дан. - STD 11, RFC 822, UDEL, 1982
9. Eastlake D., Crocker S., Schiller J. Randomness Recommendations for Security [Электронный ресурс]: — Электрон. дан. - RFC 1750, DEC, Cybercash, MIT, 1994.
10. Feller W. An Introduction to Probability Theory and its Applications - N.-Y.: John Wiley and Sons, 1968 — Т. 1.
11. Floyd S., Jacobson V. The synchronization of periodic routing messages in SIGCOMM Symposium on Communications Architectures and Protocols // Computer Communications Review. — 1993. — № 20. — С. 33–44.
12. International Standards Organization, "ISO/IEC DIS 10646-1:1993 information technology -- universal multiple-octet coded character set (UCS) -- part I: Architecture and basic multilingual plane," 1993.
13. Mills D. Network Time Protocol Version 3 [Электронный ресурс]: — Электрон. дан. - RFC 1305, UDEL, 1992.
14. Mockapetris P. Domain Names - Concepts and Facilities [Электронный ресурс]: — Электрон. дан. - STD13, RFC 1034, USC/Information Sciences Institute, 1987.
15. Mockapetris P. Domain Names - Implementation and Specification [Электронный ресурс]: — Электрон. дан. - STD 13, RFC 1035, USC/Information Sciences Institute, 1987.
16. Postel J. Internet Protocol [Электронный ресурс]: — Электрон. дан. - STD 5, RFC 791, USC/Information Sciences Institute, 1981.

17. Quality of Service for Voice over IP [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2005. — Режим доступа: http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/qossol/qosvoip.htm., свободный.
18. Odom W., Cavanaugh M. Cisco DQOS Exam Certification Guide. — Cisco Press, 2003. — 936 с.
19. Rekhter Y., Moskowitz R., Karrenberg D. Address Allocation for Private Internets [Электронный ресурс]: — Электрон. дан. - RFC 1597, T.J. Watson Research Center, IBM Corp., Chrysler Corp., RIPE NCC, 1994.
20. Reynolds J., Postel J. Assigned Numbers [Электронный ресурс]: — Электрон. дан. - STD 2, RFC 1700, USC/Information Sciences Institute, 1994.
21. The Unicode Standard. The Unicode Consortium [Электронный ресурс]: — Электрон. дан. - N.-Y.: Addison-Wesley, 1991.
22. Traffic Analysis for Voice over IP [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2005. — Режим доступа: http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/voipsol/ta_isd.htm., свободный.
23. Unified Messaging [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2005. — Режим доступа: http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/voipsol/um_isd.htm., свободный.
24. Voice Network Design Fundamentals [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2005. — Режим доступа: http://www.pluscom.ru/cisco_product/cc/td/doc/product/access/sc/rel9/soln/voip20/impl/scigdesn.htm., свободный.
25. Voydock V. L., Kent S. T. Security mechanisms in high-level network protocols [Электронный ресурс]: — Электрон. дан. - ACM Computing Surveys, vol. 15, pp. 135--171, June 1983.
26. White Paper Architecture for Voice, Video and Integrated Data [Электронный ресурс]: — Электрон. дан. — Cisco Systems Inc, 2005. — Режим доступа: http://www.cisco.com/warp/public/cc/so/neso/vvda/iptl/avvid_wp.htm., свободный.
27. White Paper IP Telephony Security: Deploying Secure IP Telephony in the Enterprise Network [Электронный ресурс]: — Электрон. дан. — META Group Inc, 2005. — 1 электрон. опт. диск (CD-ROM)

# Приложение А. Руководство программиста

Создание приложения на основе Packet Sniffer SDK состоит из пяти основных шагов:

1. Инициализация Packet Sniffer SDK;
2. Получение информации о присутствующих в системе сетевых адаптерах;
3. Выбор и открытие сетевого адаптера;
4. Создание фильтра для интересующего трафика;
5. Захват и анализ трафика.

Основным компонентом (ядром) Packet Sniffer SDK является компонент HNPSManager, ответственный за инициализацию библиотеки. HNPSManager загружает внутренний драйвер, получает информацию о сетевых адаптерах и создаёт объект класса HNAdapterConfig для каждого обнаруженного адаптера.

Инициализация Packet Sniffer SDK происходит в два этапа – во-первых – при создании экземпляра объекта формы приложения в функции FormCreate (TObject *Sender). В данной функции происходит инициализации рабочей консоли приложения HNConsole = new CHNLogConsole(), а так же инициализация рабочей очереди приложения HNAdapter->ReceiveQueue = HNQueue->Handle, обеспечивающей буферизацию пакетов поступающих от драйвера захвата и синхронизацию многопоточной работы с данным буфером. Помимо этого в данной процедуре происходит инициализация визуальных компонент интерфейса пользователя.

Во-вторых, при нажатии пользователя на кнопку инициализации (Initialize manager) запускается обработчик данного события - Init_BtnClick(TObject *Sender). В рамках данной функции выполняется инициализация HNPSManager и диагностирование в случае возникновения ошибки - Res = HNPSManager->Initialize().

Вторым этапом работы приложения является получение информации о присутствующих в системе сетевых адаптерах. Данная операция выполняется при каждом изменении конфигурации, содержащейся в HNAdapterConfig, для какого-либо из адаптеров. Это достигается путём задания обработчика события OnConfigChange для HNPSManager – процедуры HNPSManagerConfigChange(TObject *Sender, Pointer hConfig, DWORD ChangeType). Данный обработчик проверяет тип изменения конфигурации и в случае если это изменение вызвано обнаружением нового сетевого адаптера, добавляет его к списку адаптеров на главном окне приложения:

```
if(ChangeType == acCreate)
{
    // Adding to the list only adapters of the "Ethernet" type
   if(HNAdapterConfig->AdapterType == atEthernet)
   {
      cbNetCardList->Items->AddObject(HNAdapterConfig->AdapterDescription,
        (TObject*)hConfig);
   }
}
```

Открытие сетевого адаптера происходит при нажатии кнопки "Open" на графическом интерфейсе пользователя – при этом происходит вызов обработчика Open_BtnClick(TObject *Sender). Функция проверяет выбранный элемент в списке обнаруженных адаптеров, в случае если список пуст, происходит возврат без дальнейшей обработки. Если список не пуст, производится конфигурирование объекта HNAdapter в соответствии с выбранным адаптером:

```
if(cbNetCardList->ItemIndex == -1) return;
HNAdapterConfig->Handle = (void*)cbNetCardList->Items->Objects[cbNetCardList->ItemIndex];
HNAdapter->ConfigHandle = HNAdapterConfig->Handle;
```

В рамках данной функции также происходит дополнительная инициализация очереди приложения в соответствии с выбранным адаптером – задание максимальной длины пакета: HNQueue->MaxPacketSize = HNAdapterConfig->MaxPacketSize.

Так же происходит выделение памяти для очереди пакетов, вызовом метода AllocItems(), в случае неудачного выделения памяти происходит освобождение выделенных ресурсов:

```
if (HNQueue->AllocItems() != HNERR_OK)
{
   Stop();
   Application->MessageBox("Error allocate queue's packets","Error",IDOK);
   return;
}
```

При успешном выделении памяти, происходит запуск потока необходимого для работы очереди пакетов с очисткой ресурсов в случае возникновения ошибки и диагностическим сообщением:

```
if (HNQueue->Start() != HNERR_OK)
{
   Stop();
   Application->MessageBox("Error creating queue's thread","Error",IDOK);
   return;
}
```

Вывод диагностического сообщения в консоль приложения в случае успешного запуска потока происходит из обработчика события OnThreadBegin для потока очереди - HNQueueOnThreadBegin(TObject *Sender, DWORD &ThParam): HNConsole->printf("Work thread start\n").

После этого для адаптера устанавливается MAC-фильтр, пропускающий все пакеты HNAdapter->MacFilter = mfAll и производится непосредственное открытие адаптера вызовом Res = HNAdapter->OpenAdapter();

```
if(Res != HNERR_OK)
{
   Stop();
   HNGetErrorBox(Res);
   return;
}
```

При этом обеспечивается вывод диагностического сообщения и необходимая очистка в случае неудачной попытки. Очистка выполняется функцией Stop(). В случае если адаптер был успешно открыт, происходит вывод соответствующего сообщения в консоль приложения и открытие файлов для логирования полученных пакетов:

```
if(chkDumpToFile->Checked)
{
   hLogFile = CreateFile(DumpFileName->Text.c_str(),
                         GENERIC_READ|GENERIC_WRITE,
                         0, NULL, CREATE_ALWAYS,
                         FILE_ATTRIBUTE_NORMAL, 0);

   if(hLogFile == INVALID_HANDLE_VALUE)
   {
```

```
      Stop();
      Application->MessageBox("Error opening dump file.","Error",IDOK);
      return;
   }
}

if(chkDumpRTP->Checked)
{
   hRTPFile = CreateFile(DumpRTPName->Text.c_str(),
                         GENERIC_READ|GENERIC_WRITE,
                         0, NULL, CREATE_ALWAYS,
                         FILE_ATTRIBUTE_NORMAL, 0);

   if(hRTPFile == INVALID_HANDLE_VALUE)
   {
      Stop();
      Application->MessageBox("Error opening RTP dump file.","Error",IDOK);
      return;
   }
}
```

Первый файл используется для логирования проанализированных Ethernet пакетов в шестнадцатеричном представлении, во второй происходит логирование пакетов в более удобном для человека представлении – "название поля: значение".

После того как был открыт выбранный сетевой адаптер – на графическом интерфейсе пользователя становятся активными кнопки для активации фильтров. Данное приложение не содержит средств для написания собственных BPF фильтров, поскольку на практике крайне редко возникает необходимость использования фильтров, отличных от поставляемых в составе данного приложения. Для этих целей следует использовать сторонние программные продукты, сохраняя откомпилированные фильтры. В целях обеспечения большей гибкости Sniffer позволяет загружать откомпилированные BPF фильтры из файлов – соответствующая кнопка расположена на главной форме приложения. При нажатии её вызывается функция-обработчик EnableFilter_BtnClick(TObject *Sender).

Данная функция выполняет загрузку кода из файла и вывод диагностического сообщения в консоль:

```
Res = HNUserFilter->LoadFromFile(UserFilterFileName->Text.c_str());
if(Res != HNERR_OK)
{
   HNGetErrorBox(Res);
   return;
}
```

В случае успешной загрузки производится установка и активация данного фильтра для открытого сетевого адаптера:

```
HNAdapter->UserFilter = HNUserFilter->Handle;
HNAdapter->UserFilterActive = true;
```

Загруженный фильтр пользователь может деактивировать, нажав кнопку "Disable Filter" – при этом выполнится обработчик DisableFilter_BtnClick(TObject *Sender):

```
HNAdapter->UserFilterActive = false;
HNAdapter->UserFilter = NULL;
HNConsole->printf("User settable BPF filter disabled ...\n\n");
```

Также на форме приложения находится check-box, позволяющий включать использования FastBPF. Для этих целей служит функция chkUseFastBpfClick(TObject *Sender), выполняющая установку флага использования FastBPF: HNAdapter->UseFastUserFilter = chkUseFastBpf->Checked.

В соответствии с предназначением анализатора пакетов, его основная функциональность обеспечивается функцией HNQueueOnPacketReceive(TObject *Sender, ThParam, Pointer hPacket, Pointer pPacketData, int IncPacketSize). Данная функция вызывается всякий раз как в очередь поступает захваченный драйвером пакет.

Функция осуществляет ведение основной статистики, ведение расширенной статистики производится только в том случае, если не ведётся логирование захваченных пакетов:

```
dwCapTotal_Count = dwCapTotal_Count + 1;
dwCapTotal_Bytes = dwCapTotal_Bytes + DWORD(HNPacket->PacketSize);
DumpCount = DumpCount + 1;
```

Помимо статистики функция обеспечивает логирование пакетов и ограничение количества пакетов для обработки – захват прекращается автоматически при достижении определённого числа обработанных пакетов:

```
if(UseLimit)
{
   if (DumpCount <= LimitCount)
   {
      DumpPacket(hPacket);
      if(DumpCount == LimitCount)
      {
         PostMessage(Handle,HN_WM_THREAD_EXIT,0,0);
      }
   }
}
else
{
   DumpPacket(hPacket);
}
```

Вызываемая функция DumpPacket, получающая в качестве параметра указатель на пакет, вызывает функцию ProcessPacket, назначение которой – ведение статистики захваченных пакетов по протоколам. Помимо этого вызова DumpPacket, при необходимости логирования пакетов, осуществляет преобразование захваченного пакета в шестнадцатеричную или "название поля: значение" форму вызовом DumpLen = PacketToStr(DumpBuff,hPkt). Возвращаемое значение содержит число символов, при этом полученная в результате преобразования строка записывается в соответствии с заданными пользователем предпочтениями – в консоль и/или файл.

При необходимости завершения работы приложения пользователь нажимает кнопку "Close", в результате происходит вызов функции Close_BtnClick(TObject *Sender). Данная функция производит освобождение выделенных для очереди ресурсов и закрывает адаптер:

```
if (HNAdapter->IsOpened())
{
   HNAdapter->MacFilter = 0;
   Timer->Enabled = False;
   GetStat(False);
   HNAdapter->CloseAdapter();
}
```

```
if (HNQueue->IsStarted())
   HNQueue->Stop();

if (HNQueue->AllocatedSize > 0)
   HNQueue->FreeItems();
```

Также данная функция закрывает открытые для логирования файлы.

Во время работы приложения отображение статистики захваченных пакетов осуществляется при помощи таймера – при возникновении события таймера данные на основной форме обновляются:

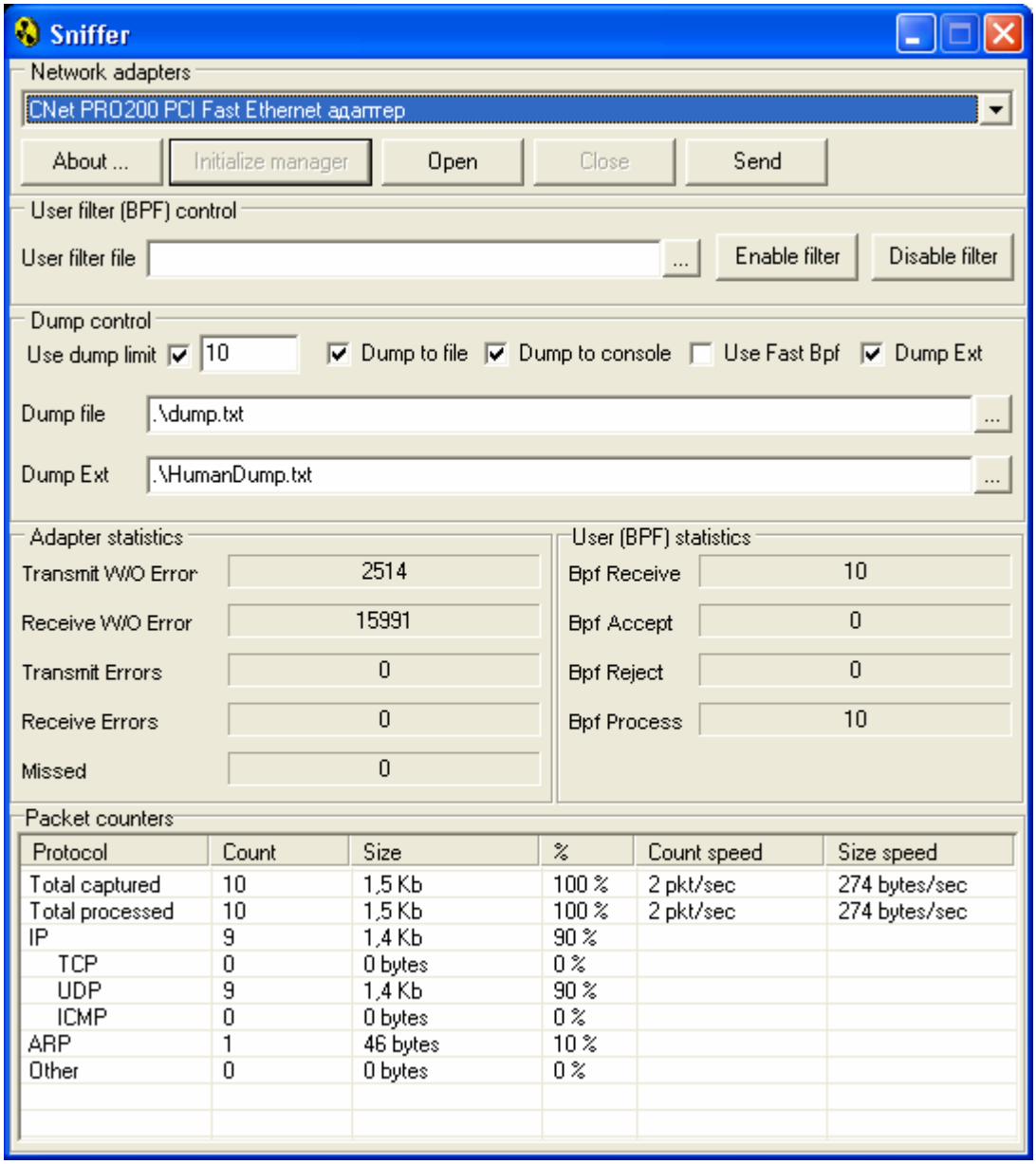
```
GetStat(false);
ShowCounters();
```

При необходимости генерации трафика данный таймер также служит для задания интенсивности генерируемого трафика.

# Приложение Б. Руководство пользователя

Приложение Sniffer представляет собой форму, отображаемую поверх остальных окон (рис. Б.1).

Рисунок Б.1 – Главное окно приложения




Для начала работы пользователю необходимо нажать кнопку инициализации - "Initialize Manager", в результате чего будет заполнен список активных сетевых адаптеров. После заполнения списка пользователь имеет возможность выбора адаптера, который будет осуществлять захват пакетов. Приложение может работать только с одним адаптером одновременно, при необходимости захватывать пакеты с нескольких сетевых адаптеров одновременно, следует открыть несколько – по числу отслеживаемых адаптеров – экземпляров приложения, каждый из которых захватывает пакеты со своего адаптера.

Выбрав адаптер из полученного списка, пользователь для начала захвата должен нажать кнопку "Open". Перед этим следует указать в какой форме и куда логировать захваченные пакеты:

- Строка "Dump to file" – задаёт путь к файлу, использующемуся для записи полученных пакетов. Если файл с указанным в строке именем не существует – он будет создан. При каждом запуске программы всё существовавшее до этого содержимое файла уничтожается, это сделано для удобства работы пользователя, поскольку присутствие избыточной информации затрудняет анализ;
- Строка "Dump Ext" – задаёт путь к файлу, использующемуся для записи полученных пакетов в удобной для человека форме. Если файл с указанным в строке именем не существует – он будет создан. При каждом запуске программы всё существовавшее до этого содержимое файла также уничтожается.
- "Use dump limit" – определяет следует ли использовать лимит на количество обработанных пакетов, при достижении которого прекращается захват пакетов;
- "Dump to file" – определяет следует ли логировать захваченные пакеты в файл в шестнадцатеричной форме;
- "Dump to console" – определяет следует ли логировать захваченные пакеты в консоль приложения;
- "Use fast bpf" – определяет следует ли приложению использовать FastBPF фильтры
- "Dump Ext" – определяет следует ли логировать захваченные пакеты в файл в форме удобной для восприятия человеком – при этом, пакеты неопознанного типа также логируются в шестнадцатеричной форме. При использовании расширенного логирования, логирование в файл стандартного дампа не производится.

При помощи контрола "User filter file" пользователь имеет возможность загружать сохранённые в файлах BPF фильтры. В состав приложения входит несколько стандартных фильтров, хранящихся в папке "bpf" рядом с исполняемым модулем. Данные фильтры позволяют фильтровать трафик по протоколу, при этом имя файла соответствует названию протокола. Активация и деактивация загруженных фильтров производится кнопками "Enable filter" и "Disable filter" соответственно.

Статистика захваченных пакетов отображается в следующих полях:

- "Transmit W/O Error" – количество кадров, переданных сетевым адаптером с момента установления связи без ошибок;
- "Receive W/O Error" – количество кадров, принятых сетевым адаптером с момента установления связи без ошибок;
- "Transmit Err" – количество кадров, переданных сетевым адаптером с момента установления связи с ошибоками;
- "Receive Err" – количество кадров, полученных сетевым адаптером с момента установления связи с ошибоками;
- "Missed" - количество кадров, пропущенных сетевым адаптером с момента установления связи.

Статистика по фильтрации данных пакетов отображается в полях:

- "Bpf Receive" – количество кадров, полученных фильтром с момента его загрузки;
- "Bpf Accept" – количество кадров, принятых фильтром с момента его загрузки;
- "Bpf Reject" – количество кадров, отвергнутых фильтром с момента его загрузки;
- "Bpf Process" – количество кадров, обработанных фильтром с момента его загрузки.

На форме также отображается статистика по основным протоколам и показатели скорости обработки поступающих пакетов.

При необходимости генерации трафика пользователь может использовать кнопку "Send", и с помощью всплывающего диалогового окна задать размер, адрес источника и получателя, а так же интенсивность генерируемых пакетов. Помимо этого всплывающая

форма обеспечивает возможность проверки безопасности сети путём проведения типовых атак.

При необходимости завершения работы пользователь должен нажать кнопку "Close", останавливая тем самым захват пакетов. При этом на форме будет отображена статистика по обработанным пакетам.

Диагностическая информация, генерируемая приложением в процессе работы, выводится в консоли приложения (рис. Б.2), помимо этой информации в консоль приложения могут выводиться захваченные приложением пакеты в установленной пользователем форме.

Рисунок Б.2 – Консоль приложения