МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

«СОГЛАСОВАНО»
Руководитель ОП
_____ Добржинский Ю.В.
«01» сентября 2017 г.

«УТВЕРЖДАЮ»
Заведующий кафедрой
«Информационные системы управления»
_____ А.И. Сухомлинов
«01» сентября 2017 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Сети и телекоммуникации

Направление подготовки 09.03.01 Информатика и вычислительная техника
Профиль «Автоматизированные системы обработки информации и управления»
Форма подготовки очная

курс 3,4 семестр 6,7
лекции 54 час.
практические занятия ___час.
лабораторные работы 90 час.
в том числе с использованием МАО лек.___/пр.___/лаб.___ час.
в том числе в электронной форме лек.___/пр.___/лаб.___ час.
всего часов аудиторной нагрузки 144 час.
в том числе с использованием МАО ___ час.
в том числе контролируемая самостоятельная работа ___ час.
в том числе в электронной форме ___ час.
самостоятельная работа 72 час.
в том числе на подготовку к экзамену 27 час.
курсовая работа / курсовой проект ___ семестр
зачет 6 семестр
экзамен 7 семестр

Рабочая программа составлена в соответствии с требованиями образовательного стандарта, самостоятельно установленного ДВФУ по направлению подготовки 09.03.01 Информатика и вычислительная техника, утвержденный приказом ректора ДВФУ от 04.04.2016 № 12-13-593

Рабочая программа обсуждена на заседании кафедры «Информационные системы управления», протокол № 1 от «1» сентября 2017 г.

Заведующий кафедрой: к.т.н., доцент Сухомлинов А.И.
Составитель: ассистент Пашин С.С.

I. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

II. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

ABSTRACT

Bachelor's degree in *09.03.01 "Computer Science and Computer Engineering"*

Study profile Program "Automated systems of information processing and management"

Course title: *Networks and Telecommunications.*

Variable part of Block 1, 6 credits

Instructor: *Pashin S.S.*

At the beginning of the course a student should be able to: *device PC, knowledge of operating systems*

Learning outcomes: *the ability to install software and hardware for information and automated systems;*

ability to develop business plans and technical assignments for equipping departments, laboratories, offices with computer and network equipment;

ability to participate in setting up and adjustment of software and hardware systems

Course description: *The content of the discipline covers the following range of issues: basic concepts of administering information systems; components of the information computer system; network and personal operating systems; active network equipment; principles of building an open system and "client-server" technologies, the ISO / OSI model; TCP / IP and its protocols; addressing in IP networks; principles and basic protocols of Internet routing; application-level protocols; programming in the Internet; concept of IT service*

Main course literature:

1. Olifer VG, Olifer N.A. Computer networks. Principles, technology, protocols: a textbook for universities. 4 th ed. - SPb .: Peter, 2013. - 944s.

2. Pozin B.A. Commissioning of information systems and maintenance of their software. - Moscow: New Technologies, 2010. - 32 c., 1 copy,

https://lib.dvfu.ru:8443/lib/item?id=chamo:292960&aid=CP84pm5146DTSbLXtycdQJG4AK2K/0/PBzsf1PhzYoE%3D%3ByeqaSYMZqTbToVxBhdcTOw%3D%3D%3Bml7M8gqATOl5vBLJ1vkaSnM7wr4g3veN9RBaQWquuFoY3wLtX1R/QjkNV6rBhbeEdthKLHmRxYCooy3sMTVh3HIAJzg4KAa9cA57OmuK28c%3D

Form of final control*: exam and pass-fail exam.*

Аннотация к рабочей программе дисциплины
«Сети и телекоммуникации»

Учебный курс «Сети и телекоммуникации» разработан для студентов 3 курса, обучающихся по направлению 09.03.01 «Информатика и вычислительная техника», в соответствие с требованиями ФГОС ВПО по данному направлению и положением об учебно-методических комплексах дисциплин образовательных программ высшего профессионального образования (утверждено приказом и.о. ректора ДВФУ от 17.04.2012 № 12-13-87).

Дисциплина «Сети и телекоммуникации» входит в блок базовой части профессиональных дисциплин (Б1.В.ОД.9).

Общая трудоемкость освоения дисциплины составляет 7 зачетных единиц, 216 часа. Учебным планом предусмотрены лекционные занятия (54 часа), лабораторные работы (90 часа) самостоятельная работа (72 часа том числе 27 часов на подготовку к экзамену). Дисциплина реализуется на 3 курсе в 6-м семестре и на 4 курсе в 7 семестре.

Содержание дисциплины охватывает следующий круг вопросов: основные понятия администрирования информационных систем; составные части информационной вычислительной системы; сетевые и персональные операционные системы; активное сетевое оборудование; принципы построения открытых системы и «клиент-серверных» технологий, модель ISO/OSI; стек TCP/IP и его протоколы; адресация в IP сетях; принципы и основные протоколы маршрутизации в Интернет; протоколы прикладного уровня; программирование в Интернет; понятие ИТ-сервиса.

Целью дисциплины является обучение принципам и методам проектирования и использования централизованных, а также распределенных сетей и систем телекоммуникаций, алгоритмам управления информационными потоками и методам технической реализации процедур передачи данных.

Задачи:

- В результате изучения курса студенты должен знать и понимать: состав и структуру инструментальных средств, тенденции их развития в части операционных систем; управление процессами (в т.ч. параллельными); взаимодействие процессов в распределенных системах.
- На основе приобретенных знаний формирование у студентов умения: инсталлировать программные компоненты информационных систем; настраивать конкретные конфигурации операционных систем; разрабатывать программы, использующие возможности операционных систем.
- Приобретение студентами навыков владения: технологиями построения и сопровождения информационных систем; приемами практической работы в среде различных операционных систем и способами их администрирования.

Результаты освоения дисциплины «Сети и телекоммуникации» достигаются за счет использования в процессе обучения: лекций с применением мультимедийных технологий, активных методов обучения; лабораторных занятий на базе компьютерной сети на платформах UNIX и Windows.

Учебная дисциплина «Сети и телекоммуникации» опирается на знания, полученные в ходе изучения курсов «Информатика», «ЭВМ и периферийные устройства», «Операционные системы».

Особенности сетевого взаимодействия информационных систем, изучаемые в рамках курса, дополняют и структурируют информацию из дисциплин «Системы реального времени», «Информационные системы управления» и «Проектирование автоматизированных систем обработки информации и управления».

Дисциплина направлена на формирование общепрофессиональных и профессиональных компетенций выпускника.

Планируемые результаты обучения по данной дисциплине (знания, умения, владения), соотнесенные с планируемыми результатами освоения образовательной программы, характеризуют этапы формирования следующих общепрофессиональных и профессиональные компетенции:

Код и формулировка компетенции	**Этапы формирования компетенции**	
ОПК-1 - способностью инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем	Знает	- основные проприетарные и свободно распространяемые сетевые программные пакеты, принципы их функционирования и взаимодействия; - концептуальные схемы работы сетевого оборудования, принципы сетевого взаимодействия; - технологии физического уровня.
	Умеет	- выбирать необходимые программные пакеты, исходя из потребностей существующей, либо разрабатываемой информационной системы; - выбирать активное и пассивное сетевое оборудование исходя из потребностей системы в сетевом взаимодействии; - настраивать сетевое оборудование; - настраивать сетевые программные пакеты.
	Владеет	- системным подходом в выборе компонентов и технологий при построении сетевой инфраструктуры автоматизированной системы.
ОПК-3 - способностью разрабатывать бизнес-планы и технические задания на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием	Знает	- принципы построения структурируемых кабельных систем; - компоненты компьютерного и сетевого оборудования; - особенности функционирования различного сетевого оборудования (серверов, маршрутизаторов, коммутаторов, модемов, телефонных станций, элементов оптических кроссовых систем);
	Умеет	- составлять техническое задание на построение СКС; - формировать требования к аппаратным системам и программным системам, исходя из потребностей предприятия.
	Владеет	- навыками планирование и проведение работ по расширению сетевой структуры предприятия; - навыками формирования требований к системе, разработкой технического задания на СКС.
ОПК-4 – способностью участвовать в настройке и наладке программно-аппаратных комплексов	Знает	- сетевые протоколы (стек TCP/IP) и их реализации; - физическое построение сетей (Ethernet, 802.11); - основные сетевые сервисы, используемые в информационных системах предприятий (корпоративная почта, телефония, СУБД, веб-сервер, прокси-сервер, бренмауер, служба каталогов); - серверное и сетевое оборудование.
	Умеет	- использовать теоретические знания сетевых протоколов, для

Код и формулировка компетенции	Этапы формирования компетенции	
		построения информационных систем; - диагностировать работоспособность сети, находить неисправности в настройке сетевых протоколов; - настраивать рабочие станции и сервера.
	Владеет	- навыками администрирования серверных операционных систем (Windows, FreeBSD); - навыками настройки сетевого оборудования Cisco.
ПК-3 - способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	Знает	- методы проектирования сетей передачи данных; - инструментальными средствами исследования работы сети;
	Умеет	- анализировать результаты, полученные при исследовании работы сети;
	Владеет	- методами и инструментальными средствами исследования, моделирования и проектирования сетей передачи данных.

Для формирования вышеуказанных компетенций в рамках дисциплины «Сети и телекоммуникации» применяются следующие методы активного/ интерактивного обучения: лекция визуализация, с применением мультимедийного оборудования (наглядные материалы, слайды, презентации), лекция-беседа, семинары.

I. СТРУКТУРА И СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОЙ ЧАСТИ КУРСА

Раздел I. Основы сетей передачи данных (14 час.)

Тема 1. Эволюция компьютерных сетей с использованием методов активного обучения – лекция-визуализация (2 час.)

Цели и задачи курса. Литература. Роль дисциплины в образовательной программе. Вычислительные и телекоммуникационные технологии. Первые компьютерные сети. Конвергенция сетей.

Тема 2. Общие принципы построения сетей с использованием методов активного обучения – лекция-визуализация (2 час.)

Простейшая сеть из двух компьютеров. Совместное использование ресурсов. Сетевое программное обеспечение. Физическая передача по линиям связи. Кодирование. Проблемы связи нескольких компьютеров. Обобщенная задача коммутации.

Тема 3. Коммутация каналов и пакетов с использованием методов активного обучения – лекция-визуализация (2 час.)

Коммутация каналов. Элементарный канал. Составной канал. Неэффективность передачи пульсирующего трафика. Коммутация пакетов. Классификация коммуникации пакетов. Сравнение сетей с коммутацией пакетов и каналов.

Тема 4. Архитектура и стандартизация сетей (2 час.)

Декомпозиция задачи сетевого взаимодействия. Многоуровневый подход. Протокол и стек протоколов. Модель OSI. Общая характеристика модели OSI. Стандартизация сетей. Понятие открытой системы. Информационные и транспортные услуги.

Тема 5. Примеры сетей с использованием методов активного обучения – лекция-визуализация (2 час.)

Классификация компьютерных сетей. Обобщенная структура телекоммуникационной сети. Сеть доступа. Магистральная сеть. Информационные центры. Сети операторов связи. Корпоративные сети. Интернет.

Тема 6. Сетевые характеристики с использованием методов активного обучения – лекция-беседа(2 час.)

Типы характеристик. Субъективные оценки качества. Характеристики и требования к сети. Производительность. Статистические оценки характеристик сети. Характеристики задержек пакетов. Надежность. Характеристики потерь пакетов. Характеристики сети поставщика услуг.

Тема 7. Методы обеспечения качества обслуживания с использованием методов активного обучения – лекция-беседа (2 час.)

Обзор методов обеспечения качества обслуживания. Приложения и качество обслуживания. Чувствительность трафика к задержкам, потерям и искажениям пакетов. Анализ очередей. Модель M/M/1. Техника управления очередями. Механизмы кондиционирования трафика. Обратная связь. Резервирование ресурсов. Инжиниринг трафика.

Раздел II Технологии физического уровня (8 час.)

Тема 1. Линии связи с использованием методов активного обучения – лекция-визуализация (2 час.)

Классификация линий связи. Первичные сети, линии и каналы связи. Физическая среда передачи данных. Аппаратура передачи данных. Характеристики линий связи. Спектральный анализ сигналов на линиях связи. Затухание и волновое сопротивление. Помехоустойчивость и достоверность. Типы кабелей.

Тема 2. Кодирование и мультиплексирование данных с использованием методов активного обучения – лекция-беседа (2 час.)

Модуляция. Комбинированные методы модуляции. Дискретизация аналоговых сигналов. Методы кодирования. Выбор способа кодирования. Обнаружение и коррекция ошибок. Методы обнаружения и коррекции ошибок. Мультиплексирование и коммутация.

Тема 3. Беспроводная передача данных с использованием методов активного обучения – лекция-визуализация (2 час.)

Беспроводная среда передачи. Преимущества беспроводных коммуникаций. Беспроводные линии связи. Беспроводные системы. Двухточечная связь. Связь одного источника и нескольких приемников. Связь нескольких источников и нескольких приемников. Типы спутниковых систем. Технология широкополосного сигнала.

Тема 4. Первичные сети (2 час.)

Сети PDH. Ограничения технологии PDH. Сети SONET/SDH. Методы обеспечения живучести сети. Сети DWDM: принципы работы; Типовые топологии; оптические мультиплексоры ввода-вывода. Сети OTN: причины и цели создания; стек протоколов OTN; коррекция ошибок.

Раздел III Локальные вычислительные сети (6 час.)

Тема 1. Технологии локальных сетей на разделяемой среде (2 час.)

Общая характеристика протоколов локальных сетей на разделяемой среде. Ethernet со скоростью 10 Мбит/с на разделяемой среде: MAC-адреса; возникновение коллизии; максимальная производительность сети. Технологии Tocken Ring и FDDI. Беспроводные локальные сети IEEE 802.11. Персональные сети и технология Bluetooth.

Тема 2. Коммутируемые сети Ethernet с использованием методов активного обучения – лекция-визуализация (2 час.)

Мост как предшественник и функциональный аналог коммутатора. Логическая структуризация сетей и мосты. Коммутаторы. Параллельная коммутация. Борьба с перегрузками. Скоростные версии Ethernet: Fast Ethernet, Gigabit Ethernet, 10G Ethernet. Архитектура коммутаторов.

Тема 3. Интеллектуальные функции коммутаторов с использованием методов активного обучения – лекция-беседа(2 час.)

Алгоритм покрывающего дерева. Классическая версия STP. Агрегирование линий связи в локальных сетях. Транки и логические каналы. Борьба с «размножением» пакетов. Фильтрация трафика. Виртуальные локальные сети. Ограничения коммутаторов.

Раздел IV Сети TCP/IP (12 час.)

Тема 1. Адресация в стеке протоколов TCP/IP с использованием методов активного обучения – лекция-беседа(2 час.)

Стек протоколов TCP/IP. Типы адресов стека TCP/IP. Формат IP-адреса. Использование масок при IP-адресации. Порядок назначения IP-адресов. Адресация и технология CIDR. Отображение IP-адресов на локальные адреса. Система DNS. Протокол DHCP.

Тема 2. Протокол межсетевого взаимодействия с использованием методов активного обучения – лекция-беседа (2 час.)

Формат IP-пакета. Схема IP-маршрутизации. Упрощенная таблица маршрутизации. Пример IP-маршрутизации без масок. Маршрутизация с использованием масок. Структуризация сети самками одинаковой длины. Просмотр таблиц маршрутизации с учетом масок. Использование масок переменной длины. Фрагментация IP-пакетов.

Тема 3. Базовые протоколы TCP/IP с использованием методов активного обучения – лекция-беседа (4 час.)

Протоколы транспортного уровня TCP и UDP. Порты и Сокеты. Повторная передача и скользящее окно. Реализация метода скользящего окна в протоколе TCP. Общие свойства и классификация протоколов маршрутизации: RIP, OSPF, BGP. Протокол ICMP.

Тема 4. Дополнительные функции маршрутизаторов IP-сетей с использованием методов активного обучения – лекция-беседа (4 час.)

Фильтрация пользовательского трафика. Стандарты QoS в IP- сетях. Алгоритм ведра маркеров. Трансляция сетевых адресов. Причины подмены адресов. Трансляция сетевых адресов и портов. Стандартная модель группового вещания IP. IPv6 как развитие стека TCP/IP: система адресации протокола IPv6; снижение нагрузки на маршрутизаторы. Классификация маршрутизаторов по областям применения.

Раздел V Технологии глобальных сетей (16 час.)

Тема 1. Транспортные услуги технологии глобальных сетей (2 час.)

Типы публичных услуг сетей операторов связи. Технология Frame Relay. Технология ATM. Виртуальные частные сети. IP в глобальных сетях: чистая IP-сеть; протоколы HDLC и PPP.

Тема 2. Технология MPLS (4 час.)

Базовые концепции и механизмы MPLS. Совмещение коммутации и маршрутизации в одном устройстве. Пути коммутации по меткам. Протокол LDP. Мониторинг состояния путей LSP. Инжиниринг трафика в MPLS. Отказоустойчивость путей MPLS.

Тема 3. Ethernet операторского класса (4 час.)

Обзор версий Ethernet операторского класса. Стандартизация Ethernet как услуги. Технология EoMPLS. Услуги VPWS и VPLS. Мосты провайдера. Магистральные мосту провайдера с поддержкой инжиниринга трафика.

Тема 4. Удаленный доступ с использованием методов активного обучения – лекция-визуализация (2 час.)

Схемы удаленного доступа. Типы клиентов и абонентских окончаний. Режим удаленного узла. Режим удаленного управления и протокол telnet. Коммутируемый аналоговый доступ, удаленный доступ через телефонную сеть. Коммутируемый доступ через ISDN. Технология ADSL.

Тема 5. Сетевые службы с использованием методов активного обучения – лекция-беседа (2 час.)

Электронная почта. Электронные сообщения. Протокол SMTP. Схема с выделенным почтовым сервером. Протоколы POP3 и IMAP. Веб-служба. Веб-клиент и веб-сервер. Протокол HTTP. Динамические веб-страницы. IP-телефония. Стандарты на основе SIP. Связь телефонных сетей через Интернет. Протокол передачи файлов. Основные модели службы FTP. Сетевое управление в IP-сетях.

Тема 6. Сетевая безопасность с использованием методов активного обучения – лекция-беседа (2 час.)

Основные понятия информационной безопасности. Типы и примеры атак: атака отказа в обслуживании; перехват и перенаправление трафика; внедрение в компьютеры вредоносных программ; спам. Методы обеспечения информационной безопасности. Классификация методов защиты. Политика безопасности. Шифрование. Аутентификация, авторизация, аудит. Антивирусная защита. Сетевые экраны. Прокси-серверы. Протоколы защищенного канала. IPSec. Сети VPN на основе шифрования.

II. СТРУКТУРА И СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ КУРСА

Лабораторные работы (__90__час.)

Лабораторная работа №1. Корректирующие коды. Задание №1 (8 час.)

Лабораторная работа №2. Корректирующие коды. Задание №2 (8 час.)

Лабораторная работа №3. Расчет маршрутов в сетях (8 час.)

Лабораторная работа №4. Сжатие данных. Метод LZ78 (6 час.)

Лабораторная работа №5. Сжатие данных. Метод Хаффмана. (6 час.)

Лабораторная работа №6. Классификация линий связи. Характеристики линий связи. Семинарное занятие. (6 час.)

Лабораторная работа №7. Поиск и исправление неисправностей в IP сети (6 час.)

Лабораторная работа №8. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Знакомство с симулятором Cisco Packet Trace (6 час.)

Лабораторная работа №9. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Основы работы с интерфейсом оборудования Cisco. (6 час.)

Лабораторная работа №10. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Настройка статической маршрутизации на оборудовании Cisco. (6 час.)

Лабораторная работа №11. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Настройка протоколов маршрутизации RIP на оборудовании Cisco (6 час.)

Лабораторная работа №12. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Применение списков доступа на оборудовании Cisco (6 час.)

Лабораторная работа №13. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Проектирование сети SOHO (8 час.)

III. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине «Сети и телекоммуникации» представлено в Приложении 1 и включает в себя:

план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;

характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

требования к представлению и оформлению результатов самостоятельной работы;

критерии оценки выполнения самостоятельной работы.

IV. КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

№	Контролируемые	Коды и этапы	Оценочные средства

п/п	разделы / темы дисциплины	формирования компетенций		текущий контроль	промежуточная аттестация
1	Раздел 1. Локальные вычислительные сети	ОПК-1	знает - основные проприетарные и свободно распространяемые сетевые программные пакеты, принципы их функционирования и взаимодействия; - концептуальные схемы работы сетевого оборудования, принципы сетевого взаимодействия; - технологии физического уровня.	Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 - 90
			Умеет - выбирать необходимые программные пакеты, исходя из потребностей существующей, либо разрабатываемой информационной системы; - выбирать активное и пассивное сетевое оборудование исходя из потребностей системы в сетевом взаимодействии; - настраивать сетевое оборудование; - настраивать сетевые программные пакеты.		

			Владеет - системным подходом в выборе компонентов и технологий при построении сетевой инфраструктуры автоматизированной системы.		
2	Раздел 2 Технологии физического уровня	ОПК-3	знает - принципы построения структурируемых кабельных систем; - компоненты компьютерного и сетевого оборудования; - особенности функционирования различного сетевого оборудования (серверов, маршрутизаторов, коммутаторов, модемов, телефонных станций, элементов оптических кроссовых систем);	Отчет по лабораторной работе №7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Задание № 6. Проектирование сети SOHO	Вопросы к экзамену 1 - 90
			Умеет - составлять техническое задание на построение СКС; - формировать требования к аппаратным системам и программным системам, исходя из потребностей предприятия.		
			Владеет - навыками планирование и		

			проведение работ по расширению сетевой структуры предприятия; - навыками формирования требований к системе, разработкой технического задания на СКС.		
3	Раздел 3. Технологии глобальных сетей	ОПК-4	знает - сетевые протоколы (стек TCP/IP) и их реализации; - физическое построение сетей (Ethernet, Token ring, FDDI, 802.11); - основные сетевые сервисы, используемые в информационных системах предприятий (корпоративная почта, телефония, СУБД, веб-сервер, прокси-сервер, бренмауер, служба каталогов); - серверное и сетевое оборудование.	Отчет по лабораторной работе №2-7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 - 90
			умеет-использовать теоретические знания сетевых протоколов, для построения информационных систем; - диагностировать работоспособность сети, находить неисправности в настройке сетевых протоколов;		

			- настраивать рабочие станции и сервера.		
			Владеет - навыками администрирования серверных операционных систем (Windows, FreeBSD); - навыками настройки сетевого оборудования Cisco.		
4	Раздел 2 Технологии физического уровня	ПК-3	знает - методы проектирования сетей передачи данных; - инструментальными средствами исследования работы сети;	Отчет по лабораторной работе №2-7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 - 90
			умеет- анализировать результаты, полученные при исследовании работы сети;		
			владеет- методами и инструментальными средствами исследования, моделирования и проектирования сетей передачи данных.		

Типовые контрольные задания, методические материалы, определяющие процедуры оценивания знаний, умений и навыков и (или) опыта деятельности, а также критерии и показатели, необходимые для оценки знаний, умений, навыков и характеризующие этапы формирования компетенций в процессе освоения образовательной программы, представлены в Приложении 2.

V. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.
2. Позин Б.А. Ввод в действие информационных систем и сопровождение их программного обеспечения. – Москва: Новые технологии, 2010. – 32 с., 1 экземпляр, https://lib.dvfu.ru:8443/lib/item?id=chamo:292960&aid=CP84pm5146DTSbLXtycdQJG4AK2K/0/PBzsf1PhzYoE%3D%3ByeqaSYMZqTbToVxBhdcTOw%3D%3D%3Bml7M8gqATOl5vBLJ1vkaSnM7wr4g3veN9RBaQWquuFoY3wLtX1R/QjkNV6rBhbeEdthKLHmRxYCooy3sMTVh3HIAJzg4KAa9cA57OmuK28c%3D
3. Фадюшин С. Г. Информатика и информационные технологии: Учеб. пособие. – Владивосток: ДВФУ, 2012. – 174 с. 78 экземпляров, https://lib.dvfu.ru:8443/lib/item?id=chamo:695338&aid=2ofKgFLF9wPyHpJQCcMzm4nKrTWyAe4SqUUzYHAlGTo%3D%3BkRvIfNYxa/0XtNGSGKTwKw%3D%3D%3Bhf1/092GjoAAnNHDaqfRrbXztRvWxK8pD0uMnvkAbp7J%2B34nTsOHVSao7FDsH%2B90kH5claU0iGZp4qpjD1aZsRzOZRChESPi%2BPyle7aGPcs%3D

Дополнительная литература

1. Никифоров С.В. Введение в сетевые технологии: Элементы применения и администрирования сетей : Учеб. пособие. - 2-е изд. - М.: Финансы и статистика, 2007. - 224 с. http://www.studentlibrary.ru/book/ISBN9785279032808.html
2. Беленькая, М.Н. Администрирование в информационных системах. [Электронный ресурс]: учебное пособие / М.Н. Беленькая, С.Т. Малиновский, Н.В. Яковенко. — Электрон. дан. — М. : Горячая линия-Телеком, 2011. — 400 с. – Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=5117
3. Wenji Mao, Fei-Yue Wang. Intelligence and Security Informatics: Research Frameworks. 2012 г. [Электронный ресурс]. Режим доступа: http://www.sciencedirect.com/science/article/pii/B9780123972002000014

Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Справочная система Cisco по сетевым протоколам и технологиям, в Wiki формате http://docwiki.cisco.com
2. Официальный сайт системы pfsense с документацией https://www.pfsense.org/
3. Официальный сайт операционной системы freeBSD с документацией https://www.freebsd.org

Перечень информационных технологий и программного обеспечения

Для обеспечения учебного процесса по дисциплине «Сети и телекоммуникации» используется следующее информационные технологии и программное обеспечение: операционная система Windows, виртуальная машина VirtualBox, программные комплексы разработанные для выполнения лабораторных работ, симулятор работы

компьютерной сети Cisco Packet Tracer, программный комплекс pfSense, Интернет, текстовый процессор MS Word, табличный процессор MS Excel.

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Методические указания к проведению теоретико-типологического анализа подборки периодической литературы по изучаемой дисциплине

Сообщения должны включать в себя библиографические списки литературы и рефераты по всем темам изучаемой дисциплины.

Список литературы должен содержать не менее 30 источников, они должны быть перечислены в алфавитном порядке, соблюдена нумерация. Список литературы должен быть оформлен по принципу реферативной работы, в обязательном порядке присутствует титульный лист и нумерация страниц. Объем работы должен составлять 10-15 страниц.

Оформление электронных ресурсов в списке литературы при ссылке на авторов выполняется согласно п.п. 4.14.1 Оформление списка литературы Процедуры ВКР ДВФУ (см. пример в Процедуре).

Оформление электронных ресурсов в списке литературы при ссылке на сайты и порталы (если не указаны авторы) рекомендуется оформлять отдельным перечнем интернет-ресурсов в общей нумерации списка литературы (в конце списка) согласно следующему примеру:

Интернет-ресурсы:

Расчёт совокупной стоимости владения (TCO). URL: http://www.akvalis.ru/service/67/. Дата обращения: 28.05.2014 г.

Методические указания к составлению глоссария

Глоссарий охватывает все узкоспециализированные термины, встречающиеся в тексте. Глоссарий должен содержать не менее 50 терминов, они должны быть перечислены в алфавитном порядке, соблюдена нумерация. Глоссарий должен быть оформлен по принципу реферативной работы, в обязательном порядке присутствует титульный лист и нумерация страниц. Объем работы должен составлять 5-10 страниц. Тщательно проработанный глоссарий помогает избежать разночтений и улучшить в целом качество всей документации. В глоссарии включаются самые частотные термины и фразы, а также все ключевые термины с толкованием их смысла. Глоссарии могут содержать отдельные слова, фразы, аббревиатуры, слоганы и даже целые предложения.

Методические указания к выполнению реферата

Цели и задачи реферата

Реферат (от лат. refero — докладываю, сообщаю) представляет собой краткое изложение проблемы практического или теоретического характера с формулировкой определенных выводов по рассматриваемой теме. Избранная студентом проблема изучается и анализируется на основе одного или нескольких источников. В отличие от курсовой работы, представляющей собой комплексное исследование проблемы, реферат направлен на анализ одной или нескольких научных работ.

Целями написания реферата являются:

- развитие у студентов навыков поиска актуальных проблем современного законодательства;
- развитие навыков краткого изложения материала с выделением лишь самых существенных моментов, необходимых для раскрытия сути проблемы;
- развитие навыков анализа изученного материала и формулирования собственных выводов по выбранному вопросу в письменной форме, научным, грамотным языком.

Задачами написания реферата являются:

- научить студента максимально верно передать мнения авторов, на основе работ которых студент пишет свой реферат;
- научить студента грамотно излагать свою позицию по анализируемой в реферате проблеме;
- подготовить студента к дальнейшему участию в научно – практических конференциях, семинарах и конкурсах;
- помочь студенту определиться с интересующей его темой, дальнейшее раскрытие которой возможно осуществить при написании курсовой работы или диплома;
- уяснить для себя и изложить причины своего согласия (несогласия) с мнением того или иного автора по данной проблеме.

Основные требования к содержанию реферата

Студент должен использовать только те материалы (научные статьи, монографии, пособия), которые имеют прямое отношение к избранной им теме. Не допускаются отстраненные рассуждения, не связанные с анализируемой проблемой. Содержание реферата должно быть конкретным, исследоваться должна только одна проблема (допускается несколько, только если они взаимосвязаны). Студенту необходимо строго придерживаться логики изложения (начать с определения и анализа понятий, перейти к постановке проблемы, проанализировать пути ее решения и сделать соответствующие выводы). Реферат должен заканчиваться выведением выводов по теме.

По своей структуре реферат состоит из:

1.Титульного листа;

2.Введения, где студент формулирует проблему, подлежащую анализу и исследованию;

3.Основного текста, в котором последовательно раскрывается избранная тема. В отличие от курсовой работы, основной текст реферата предполагает разделение на 2-3 параграфа без выделения глав. При необходимости текст реферата может дополняться иллюстрациями, таблицами, графиками, но ими не следует "перегружать" текст;

4.Заключения, где студент формулирует выводы, сделанные на основе основного текста.

5.Списка использованной литературы. В данном списке называются как те источники, на которые ссылается студент при подготовке реферата, так и иные, которые были изучены им при подготовке реферата.

Объем реферата составляет 10-15 страниц машинописного текста, но в любом случае не должен превышать 15 страниц. Интервал – 1,5, размер шрифта – 14, поля: левое — 3см, правое — 1,5 см, верхнее и нижнее — 1,5см.. Страницы должны быть пронумерованы. Абзацный отступ от начала строки равен 1,25 см.

Порядок сдачи реферата и его оценка

При оценке реферата учитываются соответствие содержания выбранной теме, четкость структуры работы, умение работать с научной литературой, умение ставить проблему и анализировать ее, умение логически мыслить, владение профессиональной терминологией, грамотность оформления.

Методические указания по дисциплине

По дисциплине изданы следующие методические указания:

- Электронные МУ для лаб. работ , «ЭМУЛЯТОР РАБОТЫ СЕТИ CISCO PACKET TRACER», Пашин С.С., Владивосток ДВФУ 2013г., 82 с.
- Электронные МУ для лаб. работ , «МЕЖСЕТЕВЫЕ ЭКРАНЫ. PFSENCE», Пашин С.С., Владивосток ДВФУ 2013г., 45 с.
- Электронные МУ для лаб. работ «КОРРЕКТИРУЮЩИЕ КОДЫ», Пашин С.С., Владивосток ДВФУ 2013г., 30 с.
- Электронные МУ для лаб. работ «РАСЧЁТ МАРШРУТОВ В СЕТЯХ», Пашин С.С., Владивосток ДВФУ 2013г., 23 с.
- Электронные МУ для лаб. работ «СЖАТИЕ ДАННЫХ», Пашин С.С., Владивосток ДВФУ 2013г., 54 с.
- Электронные МУ для лаб. работ «Утилиты сетевого уровня», Пашин С.С., Владивосток ДВФУ 2013г., 36 с.

VII. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Для обеспечения учебного процесса по дисциплине «Сети и телекоммуникации» используется следующее материально-техническое обеспечение: компьютеры, операционная система Windows, виртуальная машина VirtualBox, программные комплексы разработанные для выполнения лабораторных работ, симулятор работы компьютерной сети Cisco Packet Tracer, программный комплекс pfSense, Интернет, текстовый процессор MS Word, табличный процессор MS Excel, компьютерный класс оборудованный мельтимедийными средствами (проектор, экран), персональные компьютеры студентов.

Приложение 1

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

по дисциплине «Сети и телекоммуникации»

Направление подготовки 09.03.01 Информатика и вычислительная техника

профиль «Автоматизированные системы обработки информации и управления»

Форма подготовки очная

Владивосток
2017

План-график выполнения самостоятельной работы по дисциплине

№ п/п	Дата/сроки выполнения	Вид самостоятельной работы	Примерные нормы времени на выполнение	Форма контроля
1	Весь срок обучения	Подготовка к лекционным занятиям в активной форме	22 час.	Работа на лекционном занятии
2	12 неделя	Работа над рефератом	23 час.	Защита реферата
3	14 неделя	Подготовка к экзамену	27 час.	Экзамен

Рекомендации по самостоятельной работе студентов

Самостоятельная работа студентов, предусмотренная учебным планом, соответствует более глубокому усвоению изучаемого курса, формирует навыки исследовательской работы и ориентирует на умение применять теоретические знания на практике.

Одной из задач изучения дисциплины является создание условий для самостоятельной работы обучающихся, которая включает: самостоятельное изучение тем (разделов) дисциплины; углубленное изучение отдельных тем дисциплины с использованием дополнительной литературы и Интернет - ресурсов; возможность выполнения практических и творческих работ. Преподаватель определяет темы самостоятельной работы, ее формы и объем, разрабатывает и подбирает учебно-методическое обеспечение, составляет график консультаций, осуществляет индивидуальную педагогическую поддержку в выполнении студентом самостоятельной работы, оценивает ее результаты.

Подготовка к лекционным занятиям в активной форме

Учебным процессом предусмотрены занятия с применением методов активного обучения. В данном курсе используются три метода активного обучения, а именно лекция-визуализация, лекция беседа и работа в малых группах, задачи их следующие:

- активизация мышления, причем учащийся вынужден быть активным;
- длительное время активности — учащийся работает не эпизодически, а в течение всего учебного процесса;
- самостоятельность в выработке и поиске решений поставленных задач;
- мотивированность к обучению.

В рамках самостоятельной работы студента предусмотрена подготовка к лекциям беседам. Студентам заранее дается материал по предстоящей лекции, что бы у обучающегося сформировалось видение рассматриваемого вопроса. На занятиях преподаватель, освещая материал, постоянно контактирует с аудиторией на понимание занятия. Обращается к аудитории с вопросами касаемо материала.

Лекция-беседа, или «диалог с аудиторией», наиболее распространенная и сравнительно простая форма активного вовлечения слушателей в учебный процесс. Она предполагает непосредственный контакт преподавателя с аудиторией. Ее преимущество состоит в том, что она позволяет привлекать внимание слушателей к наиболее важным

вопросам темы, определять содержание и темп изложения учебного материала с учетом особенностей аудитории.

Трудно представить более простой способ индивидуального обучения, построенного на непосредственном контакте сторон. Эффективность этого метода в условиях группового обучения снижается из-за того, что не всегда удается вовлечь в беседу каждого из слушателей. В то же время групповая беседа позволяет расшить круг мнений сторон. Участие студентов в лекции-беседе можно обеспечить различными приемами: вопросы к аудитории, которые могут быть как элементарные, с целью сосредоточить внимание слушателей, так и проблемные.

Занятия с использованием данного метода:

- Сетевые характеристики;
- Методы обеспечения качества обслуживания;
- Кодирование и мультиплексирование данных;
- Интеллектуальные функции коммутаторов;
- Адресация в стеке протоколов TCP/IP;
- Протокол межсетевого взаимодействия;
- Базовые протоколы TCP/IP;
- Дополнительные функции маршрутизаторов IP-сетей;
- Сетевые службы с использованием;
- Сетевая безопасность.

Лабораторные занятия - Работа в малых группах; Коллективные решения лабораторных работ.

Работа в малых группах – форма активного обучения, которая дает всем обучающимся возможность участвовать в работе, практиковать навыки сотрудничества, межличностного общения. Метод наиболее подходит для проведения практических занятий по данному курсу: используются не только собственные знания, но и знания одногруппников, меняются социальные роли в зависимости от ситуации, что способствует более глубокому усвоению материала и совершенствует навыки общения и взаимодействия в группе.

Методика осуществления

Организационный этап. Подбор практического задания, отвечающего следующим критериям: не имеет однозначного и односложного решения; является практическим и полезным для учащихся; вызывает интерес у учащихся; максимально служит целям обучения.

Группа студентов делится на несколько малых групп. Количество групп определяется числом творческих заданий, которые будут обсуждаться в процессе занятия. Малые группы формируются либо по желанию студентов, либо по родственной тематике для обсуждения. В группе определяются: капитан (занимает лидирующую позицию, организует обсуждение на уровне группы, формулирует общее мнение малой группы), оппонент (внимательно слушает предлагаемые позиции во время дискуссии и формулирует вопросы по предлагаемой информации), эксперт (формирует оценочное суждение по предлагаемому решению своей малой группы и сравнивает с предлагаемыми решениями других групп).

Подготовительный этап. Каждая малая группа обсуждает творческое задание в течение отведенного времени. Задача данного этапа – сформулировать групповую позицию по заданию.

Основной этап – проведение обсуждения задания. Заслушиваются суждения, предлагаемые каждой малой группой по заданию. После каждого суждения оппоненты задают вопросы, выслушиваются ответы авторов предлагаемых позиций. В завершении формулируется общее мнение, выражающее совместную позицию по заданию.

Этап рефлексии – подведения итогов. Эксперты предлагают оценочные суждения по высказанным путям решения предлагаемых заданий осуществляют сравнительный анализ предложенного пути решения с решениями других малых групп. Преподаватель дает оценочное суждение и работе малых групп, по решению заданий, и эффективности предложенных путей решения.

Работа над рефератом

Реферат (от лат. refero — докладываю, сообщаю) представляет собой краткое изложение проблемы практического или теоретического характера с формулировкой определенных выводов по рассматриваемой теме. Избранная студентом проблема изучается и анализируется на основе одного или нескольких источников. В отличие от курсовой работы, представляющей собой комплексное исследование проблемы, реферат направлен на анализ одной или нескольких научных работ.

Целями написания реферата являются:

- развитие у студентов навыков поиска актуальных проблем современного законодательства;
- развитие навыков краткого изложения материала с выделением лишь самых существенных моментов, необходимых для раскрытия сути проблемы;
- развитие навыков анализа изученного материала и формулирования собственных выводов по выбранному вопросу в письменной форме, научным, грамотным языком.

Задачами написания реферата являются*:*

- научить студента максимально верно передать мнения авторов, на основе работ которых студент пишет свой реферат;
- научить студента грамотно излагать свою позицию по анализируемой в реферате проблеме;
- подготовить студента к дальнейшему участию в научно – практических конференциях, семинарах и конкурсах;
- помочь студенту определиться с интересующей его темой, дальнейшее раскрытие которой возможно осуществить при написании курсовой работы или диплома;
- уяснить для себя и изложить причины своего согласия (несогласия) с мнением того или иного автора по данной проблеме.

Основные требования к содержанию реферата

Студент должен использовать только те материалы (научные статьи, монографии, пособия), которые имеют прямое отношение к избранной им теме. Не допускаются отстраненные рассуждения, не связанные с анализируемой проблемой. Содержание реферата должно быть конкретным, исследоваться должна только одна проблема (допускается несколько, только если они взаимосвязаны). Студенту необходимо строго придерживаться логики изложения (начать с определения и анализа понятий, перейти к постановке проблемы, проанализировать пути ее решения и сделать соответствующие выводы). Реферат должен заканчиваться выведением выводов по теме.

По своей структуре реферат состоит из:

1.Титульного листа*;*

2.Введения, где студент формулирует проблему, подлежащую анализу и исследованию;

3.Основного текста, в котором последовательно раскрывается избранная тема. В отличие от курсовой работы, основной текст реферата предполагает разделение на 2-3 параграфа без выделения глав. При необходимости текст реферата может дополняться иллюстрациями, таблицами, графиками, но ими не следует "перегружать" текст;

4. Заключения, где студент формулирует выводы, сделанные на основе основного текста.

5.Списка использованной литературы. В данном списке называются как те источники, на которые ссылается студент при подготовке реферата, так и иные, которые были изучены им при подготовке реферата.

Объем реферата составляет 10-15 страниц машинописного текста, но в любом случае не должен превышать 15 страниц. Интервал – 1,5, размер шрифта – 14, поля: левое — 3см, правое — 1,5 см, верхнее и нижнее — 1,5 см.. Страницы должны быть пронумерованы. Абзацный отступ от начала строки равен 1,25 см.

Порядок сдачи реферата и его оценка

Реферат пишется студентами в течение триместра в сроки, устанавливаемые преподавателем по конкретной дисциплине, и сдается преподавателю, ведущему дисциплину.

По результатам проверки студенту выставляется определенное количество баллов, которое входит в общее количество баллов студента, набранных им в течение триместра. При оценке реферата учитываются соответствие содержания выбранной теме, четкость структуры работы, умение работать с научной литературой, умение ставить проблему и анализировать ее, умение логически мыслить, владение профессиональной терминологией, грамотность оформления.

Критерии оценки самостоятельной работы изложены в Приложении 2.

Приложение 2

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине «Сети и телекоммуникации»

Направление подготовки 09.03.01 Информатика и вычислительная техника

профиль «Автоматизированные системы обработки информации и управления»

Форма подготовки очная

Владивосток
2017

Паспорт ФОС

Код и формулировка компетенции	Этапы формирования компетенции	
ОПК-1 - способностью инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем	Знает	- основные проприетарные и свободно распространяемые сетевые программные пакеты, принципы их функционирования и взаимодействия; - концептуальные схемы работы сетевого оборудования, принципы сетевого взаимодействия; - технологии физического уровня.
	Умеет	- выбирать необходимые программные пакеты, исходя из потребностей существующей, либо разрабатываемой информационной системы; - выбирать активное и пассивное сетевое оборудование исходя из потребностей системы в сетевом взаимодействии; - настраивать сетевое оборудование; - настраивать сетевые программные пакеты.
	Владеет	- системным подходом в выборе компонентов и технологий при построении сетевой инфраструктуры автоматизированной системы.
ОПК-3 - способностью разрабатывать бизнес-планы и технические задания на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием	Знает	- принципы построения структурируемых кабельных систем; - компоненты компьютерного и сетевого оборудования; - особенности функционирования различного сетевого оборудования (серверов, маршрутизаторов, коммутаторов, модемов, телефонных станций, элементов оптических кроссовых систем);
	Умеет	- составлять техническое задание на построение СКС; - формировать требования к аппаратным системам и программным системам, исходя из потребностей предприятия.
	Владеет	- навыками планирование и проведение работ по расширению сетевой структуры предприятия; - навыками формирования требований к системе, разработкой технического задания на СКС.
ОПК-4 – способностью участвовать в настройке и наладке программно-аппаратных комплексов	Знает	- сетевые протоколы (стек TCP/IP) и их реализации; - физическое построение сетей (Ethernet, 802.11); - основные сетевые сервисы, используемые в информационных системах предприятий (корпоративная почта, телефония, СУБД, веб-сервер, прокси-сервер, бренмауер, служба каталогов); - серверное и сетевое оборудование.
	Умеет	- использовать теоретические знания сетевых протоколов, для построения информационных систем; - диагностировать работоспособность сети, находить неисправности в настройке сетевых протоколов; - настраивать рабочие станции и сервера.
	Владеет	- навыками администрирования серверных операционных систем (Windows, FreeBSD); - навыками настройки сетевого оборудования Cisco.
ПК-3 - способностью разрабатывать компоненты аппаратно-программных комплексов	Знает	- методы проектирования сетей передачи данных; - инструментальными средствами исследования работы сети;
	Умеет	- анализировать результаты, полученные при исследовании

и баз данных, используя современные инструментальные средства и технологии программирования		работы сети;
	Владеет	- методами и инструментальными средствами исследования, моделирования и проектирования сетей передачи данных.

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций		Оценочные средства	
				текущий контроль	промежуточная аттестация
1	Раздел 1. Локальные вычислительные сети	ОПК-1	знает - основные проприетарные и свободно распространяемые сетевые программные пакеты, принципы их функционирования и взаимодействия; - концептуальные схемы работы сетевого оборудования, принципы сетевого взаимодействия; - технологии физического уровня.	Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 - 90
			Умеет - выбирать необходимые программные пакеты, исходя из потребностей существующей, либо разрабатываемой информационной системы; - выбирать активное и пассивное сетевое оборудование исходя из потребностей системы в сетевом взаимодействии;		

			- настраивать сетевое оборудование; - настраивать сетевые программные пакеты.		
			Владеет - системным подходом в выборе компонентов и технологий при построении сетевой инфраструктуры автоматизированной системы.		
2	Раздел 2 Технологии физического уровня	ОПК-3	знает - принципы построения структурируемых кабельных систем; - компоненты компьютерного и сетевого оборудования; - особенности функционирования различного сетевого оборудования (серверов, маршрутизаторов, коммутаторов, модемов, телефонных станций, элементов оптических кроссовых систем);	Отчет по лабораторной работе №7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Задание № 6. Проектирование сети SOHO	Вопросы к экзамену 1 - 90
			Умеет - составлять техническое задание на построение СКС; - формировать требования к аппаратным системам и		

			программным системам, исходя из потребностей предприятия.		
			Владеет - навыками планирование и проведение работ по расширению сетевой структуры предприятия; - навыками формирования требований к системе, разработкой технического задания на СКС.		
3	Раздел 3. Технологии глобальных сетей	ОПК-4	знает - сетевые протоколы (стек TCP/IP) и их реализации; - физическое построение сетей (Ethernet, Token ring, FDDI, 802.11); - основные сетевые сервисы, используемые в информационных системах предприятий (корпоративная почта, телефония, СУБД, веб-сервер, прокси-сервер, бренмауер, служба каталогов); - серверное и сетевое оборудование.	Отчет по лабораторной работе №2-7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 - 90
			умеет- использовать теоретические знания сетевых протоколов, для построения информационных		

			систем; - диагностировать работоспособность сети, находить неисправности в настройке сетевых протоколов; - настраивать рабочие станции и сервера.		
			Владеет - навыками администрирования серверных операционных систем (Windows, FreeBSD); - навыками настройки сетевого оборудования Cisco.		
4	Раздел 2 Технологии физического уровня	ПК-3	знает – методы проектирования сетей передачи данных; - инструментальными средствами исследования работы сети;	Отчет по лабораторной работе №2-7. Симуляция сети передачи данных с помощью Cisco Packet Tracer. Отчет по лабораторной работе №8. Межсетевые экраны. pfSence	Вопросы к экзамену 1 – 90
			умеет- анализировать результаты, полученные при исследовании работы сети;		
			владеет- методами и инструментальными средствами исследования, моделирования и проектирования сетей передачи данных.		

Шкала оценивания уровня сформированности компетенций

Код и формулировка компетенции	**Этапы формирования компетенции**		**критерии**	**показатели**
ОПК-1 - способностью инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем	знает (пороговый уровень)	- основные проприетарные и свободно распространяемые сетевые программные пакеты, принципы их функционирования и взаимодействия; - концептуальные схемы работы сетевого оборудования, принципы сетевого взаимодействия; - технологии физического уровня.	- знание основных понятий по видам автоматизируемой деятельности и функциональным областям деятельности предприятий; - знание методологий построения информационных систем, основных компонентов аппаратных и программных комплексов	- способность перечислить и описать виды деятельности, обладающие общими свойствами с точки зрения автоматизации; - способность перечислить и описать технологии физического уровня; способность перечислить и описать компоненты ИС; - способность назвать, описать и дать характеристику существующих программным пакетам.
	умеет (продвинутый)	- выбирать необходимые программные пакеты, исходя из потребностей существующей, либо разрабатываемой информационной системы; - выбирать активное и пассивное сетевое оборудование исходя из потребностей системы в сетевом взаимодействии; - настраивать сетевое оборудование; - настраивать сетевые программные пакеты.	умение проводить интервью, осуществлять обследование предприятия, стратегическое планирование информационной системы; -осуществлять анализ и проектирование сетевых топологий и подходов построения сетей; умение осуществлять выбор оборудования и программных средств умение производить настройку оборудования в соответствии с выбранными технологиями	- способность воспринимать и фиксировать информацию о деятельности предприятия; - способность осуществлять стратегическое планирование информационной системы, проводить анализ требований; - способность структурировать требования; - способность применять методологии и методы моделирования компонентов информационной системы
	владеет (высокий)	- системным подходом в выборе компонентов и технологий при построении сетевой инфраструктуры автоматизированно	владеет современными технологиями, требуемыми для создания системной инфраструктуры информационной	- способность осуществить выбор системной технологической платформы для решения

		й системы.	системы; в курсе современных тенденций в области операционных систем, и платформ прикладных задач; инструментальными средствами реализации приложений;	практических задач; способность осуществить выбор аппаратной платформы для решения практических задач; - способность инсталлировать все компоненты информационной системы с представлением документации на все полученные промежуточные и окончательный результат.
ОПК-3 - способностью разрабатывать бизнес-планы и технические задания на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием	знает (пороговый уровень)	- принципы построения структурируемых кабельных систем; - компоненты компьютерного и сетевого оборудования; - особенности функционирования различного сетевого оборудования (серверов, маршрутизаторов, коммутаторов, модемов, телефонных станций, элементов оптических кроссовых систем);	- знание основных понятий охватывающих структурированные кабельные системы; - знание основных понятий по видам автоматизируемой деятельности и функциональным областям деятельности предприятий - знание основных характеристик сетевого оборудования и компьютеров;	- способность перечислить и описать компоненты структурированных кабельных систем, их назначение; - способность перечислить и описать порядок работ при планировании, проектировании и развертывании структурированных кабельных систем; - виды деятельности, обладающие общими свойствами с точки зрения автоматизации; - способность перечислить и описать сетевое оборудование и его характеристики;
	умеет (продвинутый)	- составлять техническое задание на построение СКС; - формировать требования к аппаратным системам и программным системам, исходя из потребностей предприятия.	умение проводить интервью, осуществлять обследование предприятия, стратегическое планирование информационной системы; -осуществлять анализ и проектирование сетевых топологий и подходов построения сетей;	- способность воспринимать и фиксировать информацию о деятельности предприятия; - способность осуществлять стратегическое планирование информационной системы, проводить анализ требований; - способность структурировать

			умение осуществлять выбор оборудования и программных средств	требования;
	владеет (высокий)	- навыками планирование и проведение работ по расширению сетевой структуры предприятия; - навыками формирования требований к системе, разработкой технического задания на СКС.	владеет современными технологиями, требуемыми для создания системной инфраструктуры информационной системы; в курсе современных тенденций в области систем передачи данных, архитектур компьютеров; владеет системным подходом в выборе необходимых комплексных решений, исходя из потребностей предприятия в программных и аппаратных ресурсах в рамках текущих ограничений;	- способность поэтапно планировать работы в решениях практических задач; способность осуществить выбор аппаратной платформы для решения практических задач; - способность формулировать, структурировать бизнес-планы и технические задания;
ОПК-4 – способностью участвовать в настройке и наладке программно-аппаратных комплексов	знает (пороговый уровень)	- сетевые протоколы (стек TCP/IP) и их реализации; - физическое построение сетей (Ethernet, 802.11); - основные сетевые сервисы, используемые в информационных системах предприятий (корпоративная почта, телефония, СУБД, веб-сервер, прокси-сервер, бренмауер, служба каталогов); - серверное и сетевое оборудование.	- знание основных понятий по видам автоматизируемой деятельности и функциональным областям деятельности предприятий; - знание методологий построения информационных систем, основных компонентов аппаратных и программных комплексов	- способность перечислить и описать сетевые протоколы и их реализации; - способность перечислить и описать основные сервисы и их реализации по видам деятельности; - способность перечислить и описать технологии физического уровня; -способность перечислить и описать компоненты ИС; - способность назвать, описать и дать характеристику существующих программным пакетам.

	умеет (продвинутый)	- использовать теоретические знания сетевых протоколов, для построения информационных систем; - диагностировать работоспособность сети, находить неисправности в настройке сетевых протоколов; - настраивать рабочие станции и сервера.	-осуществлять анализ и проектирование сетевых топологий и подходов построения сетей; -умение осуществлять выбор оборудования и программных средств умение производить настройку оборудования в соответствии с выбранными технологиями -владение комплексными подходами в вопросах диагностирования неполадок в сетевых протоколах и конфигурациях программно-аппаратных средств	- способность воспринимать и фиксировать информацию о деятельности аппаратно программного комплекса; - производить мониторинг основных характеристик аппаратно-программного комплекса; - способность структурировать требования; - знание протоколов и утилит для диагностирования работоспособности аппаратно-программных комплексов; -применение концептуальных теоретических знаний для конфигурирования аппаратно-программных комплексов разных производителей
	владеет (высокий)	- навыками администрирования серверных операционных систем (Windows, FreeBSD); - навыками настройки сетевого оборудования Cisco.	владеет современными технологиями, требуемыми для создания системной инфраструктуры информационной системы; в курсе современных тенденций в области операционных систем, и платформ прикладных задач; инструментальными средствами конфигурирования оборудования;	Знает особенности функционирования различных операционных систем; Способен разворачивать базовые сервисы на оборудовании под управление различных операционных систем; - способность конфигурировать компоненты информационной системы, используя документацию по системе. -владеет базовой методологией сетевого конфигурирования аппаратно-программных средств

ПК-3 - способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	знает (пороговый уровень)	- методы проектирования сетей передачи данных; - инструментальным и средствами исследования работы сети;	- знание основных понятий по видам автоматизируемой деятельности и функциональным областям деятельности предприятий; - знание методологий построения информационных систем, основных компонентов аппаратных и программных комплексов	- способность перечислить и описать виды деятельности, обладающие общими свойствами с точки зрения автоматизации; - способность перечислить и описать технологии физического уровня; способность перечислить и описать компоненты ИС; - способность назвать, описать и дать характеристику существующих программным пакетам.
	умеет (продвинутый)	- анализировать результаты, полученные при исследовании работы сети;	-осуществлять анализ и проектирование сетевых топологий и подходов построения сетей; -умение осуществлять выбор оборудования и программных средств -владение комплексными подходами в вопросах диагностирования неполадок в сетевых протоколах и конфигурациях программно-аппаратных средств	- способность воспринимать и фиксировать информацию о деятельности предприятия; - способность осуществлять стратегическое планирование информационной системы, проводить анализ требований; - способность структурировать требования; - способность применять методологии и методы моделирования компонентов информационной системы
	владеет (высокий)	- методами и инструментальным и средствами исследования, моделирования и проектирования сетей передачи данных.	владеет современными технологиями, требуемыми для создания системной инфраструктуры информационной системы; в курсе современных тенденций в области операционных систем, и платформ	- способность осуществить выбор системной технологической платформы для решения практических задач; способность осуществить выбор аппаратной платформы для решения практических задач;

			прикладных задач; инструментальными средствами конфигурирования оборудования;	- способность инсталлировать все компоненты информационной системы с представлением документации на все полученные промежуточные и окончательный результат.

Оценочные средства для промежуточной аттестации

Экзаменационный билет состоит из 3 вопросов. Первый является общим теоретическим, подразумевает знание студентами теоретической базы построения компьютерных сетей. Второй вопрос охватывает теоретический и практический материал курса, отражает особенности конкретной технологии, понимание студентом принципов технологии и умение её использовать. Третий вопрос является практическим, содержит в себе реализацию практического задания на базе эмулятора Cisco Packet tracer. Может охватывать значительный перечень сетевых технологий, которыми студент должен владеть и уметь применять.

А) основные понятия и определения
1. Типы коммутации
2. Типы кабелей
3. Топология физических связей
4. Адресация узлов сети
5. Модель OSI. Общая характеристика модели OSI
6. Протоколы транспортного уровня TCP и UDP. Порты.
7. Трансляция сетевых адресов (NAT)

Б) основной вопрос
1. IP адресация и IPv4
2. IP адресация и IPv6
3. Операционная система сетевого взаимодействия Cisco (IOS)
4. Спам. Методы борьбы со спамом.
5. Основные концепции и настройка коммутации
6. Коммутируемые сети
7. Виртуальные частные сети (VLAN)
8. Разделение IP-сетей на подсети
9. Статическая маршрутизация
10. Динамическая маршрутизация
11. Списки контроля доступа (ACL)

В) основной вопрос
Оценка практических навыков

Оценочные средства для текущей аттестации

Текущая аттестация студентов. Текущая аттестация студентов по дисциплине «Сети и телекоммуникации» проводится в соответствии с локальными нормативными актами ДВФУ и является обязательной.

Текущая аттестация по дисциплине «Сети и телекоммуникации» проводится в форме контрольных мероприятий (работа на семинарских занятиях, выполнение практических заданий, доклад, сообщение) по оцениванию фактических результатов обучения студентов и осуществляется ведущим преподавателем.

Объектами оценивания выступают:

- учебная дисциплина (активность на занятиях, своевременность выполнения различных видов заданий, посещаемость всех видов занятий по аттестуемой дисциплине);
- степень усвоения теоретических знаний;
- уровень овладения практическими умениями и навыками по всем видам учебной работы;
- результаты самостоятельной работы.

Краткая характеристика оценочных средств:

- УО-1 - Собеседование - средство контроля, организованное как специальная беседа преподавателя с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний, обучающегося по определенному разделу, теме, проблеме и т.п.УО-3 - Доклад, сообщение - продукт самостоятельной работы обучающегося, представляющий собой публичное выступление по представлению полученных результатов решения определенной учебно-практической, учебно-исследовательской или научной темы
- УО-4 - Круглый стол, дискуссия, полемика, диспут, дебаты - оценочные средства, позволяющие включить обучающихся в процесс обсуждения спорного вопроса, проблемы и оценить их умение аргументировать собственную точку зрения.
- ПР-1 – Тест – система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающихся.
- ПР-11 - Разноуровневые задачи - реконструктивного уровня, позволяющие оценивать и диагностировать умения синтезировать, анализировать, обобщать фактический и теоретический материал с формулированием конкретных выводов, установлением причинно-следственных связей; творческого уровня, позволяющие оценивать и диагностировать умения, интегрировать знания различных областей, аргументировать собственную точку зрения.

Критерии оценки устных ответов

100-85 баллов - если ответ показывает прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение монологической речью, логичность и последовательность ответа; умение приводить примеры современных проблем изучаемой области.

85-76 - баллов - ответ, обнаруживающий прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение монологической речью, логичность и последовательность ответа. Однако допускается одна - две неточности в ответе.

75-61 - балл – оценивается ответ, свидетельствующий в основном о знании процессов изучаемой предметной области, отличающийся недостаточной глубиной и полнотой раскрытия темы; знанием основных вопросов теории; слабо сформированными навыками анализа явлений, процессов, недостаточным умением давать аргументированные ответы и приводить примеры; недостаточно свободным владением монологической речью, логичностью и последовательностью ответа. Допускается несколько ошибок в содержании ответа; неумение привести пример развития ситуации, провести связь с другими аспектами изучаемой области.

60-50 баллов – ответ, обнаруживающий незнание процессов изучаемой предметной области, отличающийся неглубоким раскрытием темы; незнанием основных вопросов теории, несформированными навыками анализа явлений, процессов; неумением давать аргументированные ответы, слабым владением монологической речью, отсутствием логичности и последовательности. Допускаются серьезные ошибки в содержании ответа; незнание современной проблематики изучаемой области.

Критерии оценки письменных ответов

100-86 баллов - если ответ показывает глубокое и систематическое знание всего программного материала и структуры конкретного вопроса, а также основного содержания и новаций лекционного курса по сравнению с учебной литературой. Студент демонстрирует отчетливое и свободное владение концептуально-понятийным аппаратом, научным языком и терминологией соответствующей научной области. Знание основной литературы и знакомство с дополнительно рекомендованной литературой. Логически корректное и убедительное изложение ответа.

85-76 - баллов - знание узловых проблем программы и основного содержания лекционного курса; умение пользоваться концептуально-понятийным аппаратом в процессе анализа основных проблем в рамках данной темы; знание важнейших работ из списка рекомендованной литературы. В целом логически корректное, но не всегда точное и аргументированное изложение ответа.

75-61 - балл – фрагментарные, поверхностные знания важнейших разделов программы и содержания лекционного курса; затруднения с использованием научно-понятийного аппарата и терминологии учебной дисциплины; неполное знакомство с рекомендованной литературой; частичные затруднения с выполнением предусмотренных программой заданий; стремление логически определенно и последовательно изложить ответ.

60-50 баллов – незнание, либо отрывочное представление о данной проблеме в рамках учебно-программного материала; неумение использовать понятийный аппарат; отсутствие логической связи в ответе.

Критерии выставления оценки на экзамене по дисциплине «Сети и телекоммуникации»

Баллы рейтинговой оценки	Оценка экзамена (стандартная)	Требования к сформированным компетенциям
От 86% до 100%	«Отлично»	Выставляется студенту, если он глубоко и прочно усвоил программный материал

Баллы рейтинговой оценки	Оценка экзамена (стандартная)	Требования к сформированным компетенциям
		курса, четко и последовательно излагает его, умеет выбирать и использовать алгоритмы планирования процессов; применять необходимые средства межпроцессного взаимодействия; владеет современными методиками проектирования, разработки х систем реального времени.
От 76% до 85%	«Хорошо»	Выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов и задач, владеет необходимыми навыками и приемами их выполнения.
От 61% до 75%	«Удовлетворительно»	Выставляется студенту, если он имеет знания только основного материала, но не усвоил его деталей, допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении программного материала, испытывает затруднения при выполнении практических заданий.
Менее 61%	«Неудовлетворительно»	Выставляется студенту, который не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические задания.

Приложение 3

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по дисциплине «Сети и телекоммуникации »

Направление подготовки 09.03.01 Информатика и вычислительная техника

профиль «Автоматизированные системы обработки информации и управления»

Форма подготовки очная

Владивосток
2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК
Кафедра информационных систем управления

С. С. Пашин

УТИЛИТЫ СЕТЕВОГО УРОВНЯ

Методические указания к лабораторной работе
по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток
2013

Приведены основные инструменты, работающие на сетевом уровне модели OSI, рассматривается структура IP пакетов. Цель - изучить инструменты, работающие на сетевом уровне модели OSI, помогающие обеспечить взаимодействие устройств в TCP/IP-сети.

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

Утилиты сетевого уровня

Рассмотрим несколько инструментов, которые используются почти каждый день практически в каждой TCP/IP-сети мира и помогают сетевому уровню решать его задачу по сквозной маршрутизации пакетов в объединенной сети:

- Протокол преобразования адресов (Address Resolution Protocol - ARP);
- Система доменных имен (Domain Name System);
- Протокол динамического конфигурирования узла (Dynamic Host Configuration Protocol - DHCP);
- Утилита ping.

Протокол преобразования адресов и система доменных имен

Конструкторы сетей должны стараться сделать использование сети как можно проще. Главным образом пользователи запоминают имя другого компьютера, к которому они хотят подключиться, так же как они помнят веб-сайт. Они определенно не хотят запомнить ни IP-адрес, ни MAC-адрес. Поэтому TCP/IP нужен протокол, который динамически определяет всю необходимую информацию, позволяя обмениваться данными, не заставляя при этом пользователя запоминать что-то кроме имени узла.

Может показаться, что запоминать имя другого компьютера не нужно. Например, в браузерах многих пользователей настроены домашние страницы по умолчанию, которые открываются сразу после запуска браузера. Можно подумать, что строка универсального указателя ресурса (Universal Resource Locator - URL) не является именем. На самом деле URL домашней страницы содержит имя этой страницы. Например, в URL http://www.cisco.com/go/precenter часть www.cisco.com представляет собой имя веб-сервера корпорации Cisco. Поэтому всякий раз, вводя имя другого компьютера или видя его на экране, пользователь обычно идентифицирует удаленный компьютер по имени.

Таким образом, протоколу TCP/IP нужен способ, который позволит компьютеру определять IP-адрес другого компьютера по его имени. Так же нужен способ определять MAC-адреса, связанные с другими компьютерами в то же локальной подсети. Схематически эта проблема показана на рис.1.

В данном случае компьютер А должен обмениваться данными с сервером, который установлен на компьютер В. Компьютер А знает своё имя, свой IP-адрес и MAC-адрес. Компьютер А не знает IP- и MAC-адреса компьютера В. Для определения этих адресов Компьютер А использует службу DNS, что бы найти IP- адрес компьютера В, и протокол ARP, чтобы определять MAC-адрес этого компьютера.

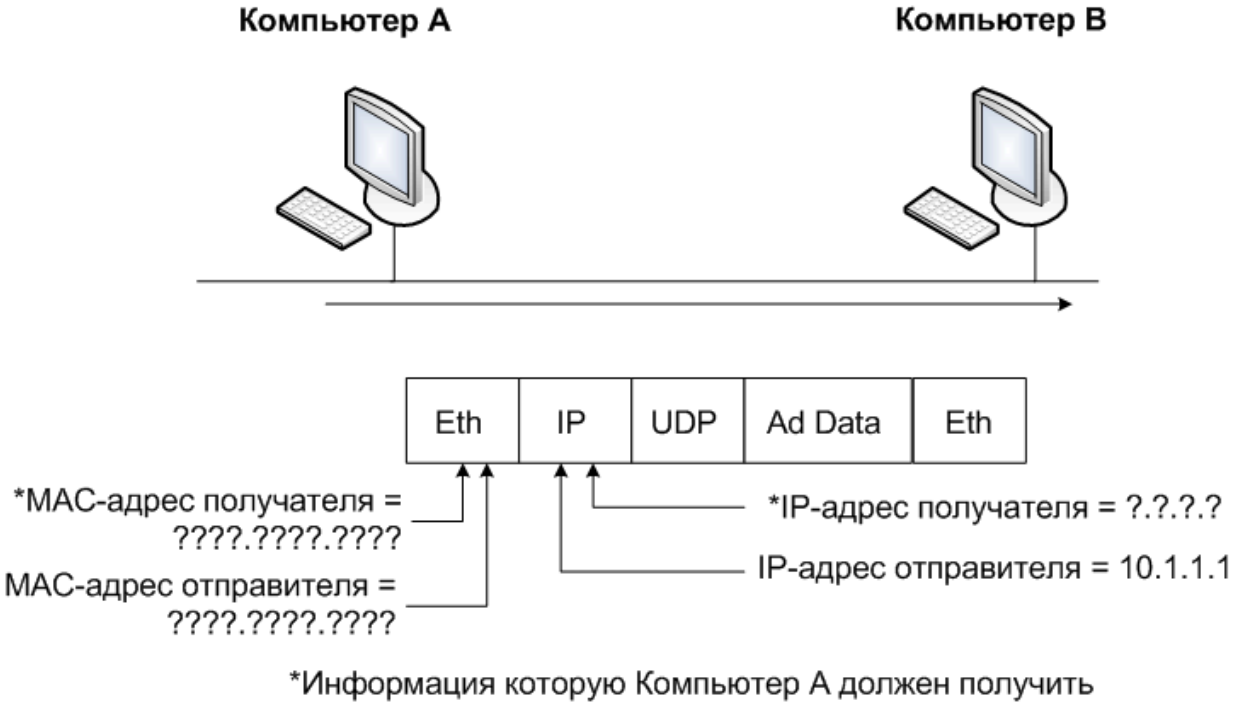


Рис.1 Компьютер А знает имя компьютера В, но нужно получить ещё IP-адрес и MAC-адрес этого компьютера

DNS-преобразование имен

Компьютер А знает IP-адрес DNS-сервера, потому что этот адрес либо заранее записан в конфигурации PC-1, либо он был получен с помощью DHCP. Как только пользователь PC-1так или иначе идентифицирует имя другого компьютера (например, jassie.example.com)компьютер отправляет DNS-серверу *запрос* на IP-адрес компьютера В. DNS-сервер в ответ отправляет адрес, 10.1.1.2. Этот простой процесс показан на рис. 2.

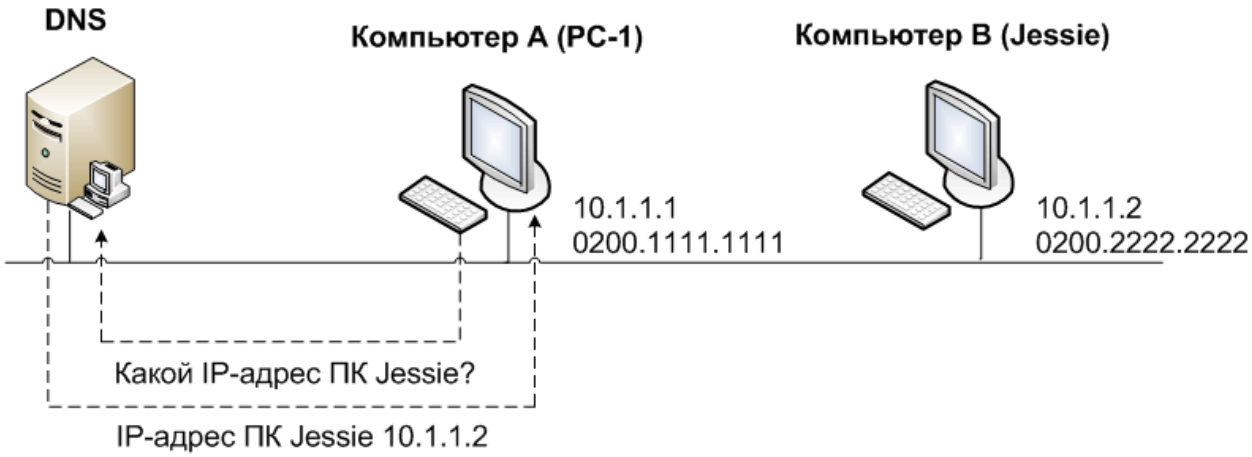


Рис.2 DNS-запрос и ответ

Компьютер А просто отправляет серверу DNS-запрос, указывая в этом запросе имя jassie или jassie.example.com, DNS-сервер в ответ отправляет IP-адрес (в данном случае 10.1.1.2). Фактически то же происходит, когда вы подключаетесь к какому-нибудь веб-сайту. Ваш ПК отправляет запрос, такой же как компьютер А отправляет для того, что бы DNS-сервер преобразовал имя компьютера В в IP-адрес. После этого ПК начинает запрашивать веб-страницу.

ARP-запрос

Как только узел узнает IP-адрес другого узла, узлу-отправителю может понадобиться МПС-адрес другого компьютера. Например, компьютер А должен знать Ethernet MAC-адрес, который соответствует IP-адресу 10.1.1.2, поэтому компьютер А начинает *ARP-рассылку (ARP broadcast)*. ARP-рассылка отправляется на широковещательный Ethernet-адрес, поэтому её получает каждый компьютер локальной сети. Поскольку компьютер В находится в той же локальной сети, что и компьютер А, он получает ARP-рассылку. Поскольку IP-адрес компьютера В 10.1.1.2, а целью ARP-рассылки является поиск MAC-адреса, связанного с этим IP-адресом, компьютер В отвечает, отправляя свой MAC-адрес. Этот процесс показан на рис.3.

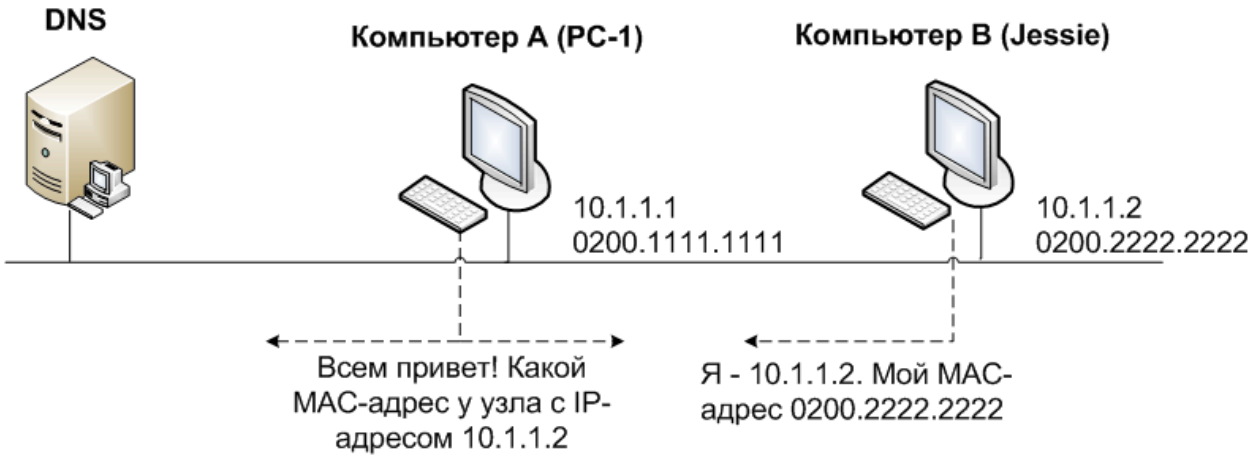


Рис. 3 Пример ARP-запроса

Теперь компьютер A знает IP-адрес получателя и Ethernet-адрес, который нужно использовать для отправки фреймов компьютеру B, и пакет, показанный на рис.1, можно успешно отправить.

Для определения MAC-адреса получателя узлы могут использовать, а могут и не использовать протокол ARP. Это зависит от логики двухэтапной маршрутизации, которую использует узел. Если узел-получатель находится в той е подсети, то узел-отправитель посылает ARP-запрос, пытаясь определить MAC-адрес получателя, как показано на рис.3. Однако если получатель и отправитель находятся в разных подсетях, то в результате логики маршрутизации отправляющего узла пакет нужно перенаправить на стандартный шлюз отправителя. Например, если компьютеры A и B расположены в разных подсетях (см.рис.1-3), логика маршрутизации компьютера A заставляет этот компьютер отправлять пакеты на стандартный шлюз (маршрутизатор). В таком случае компьютер A, использует ARP, будет определять MAC-адреса маршрутизатора вместо MAC-адреса компьютера B.

Кроме того, узлам приходится использовать ARP для определения MAC-адресов только однажды. Любое устройство, использующее протокол IP, должно сохранить или кэшировать информацию, которая была получена с помощью ARP.поместив эту информацию в свой ARP-кэш. Каждый раз, собираясь отправить пакет, инкапсулированный в Ethernet-фрейм, узел сначала проверяет свой ARP-кэш и использует найденный там MAC-адрес. Если в ARP-кэше корректная информация

отсутствует, узел может использовать ARP для определения MAC-адреса по известному IP-адресу. Кроме того, узел получает ARP-информацию, получая ARP-запрос. Например, ARP-процесс, показанный на рис.3. приводит к тому что компьютер А и компьютер В узнают MAC-адреса друг друга.

Содержимое ARP-кэша можно посмотреть в большинстве операционных систем, используя в командной строке команду ***arp -a.***

Назначение адресов и протокол DHCP

Правильный IP-адрес нужен каждому устройству, использующему TCP/IP, а точнее каждому интерфейсу в каждом устройстве, которое использует стек TCP/IP. Для некоторых устройств адрес можно и нужно задавать статически. Например, большинство широко распространенных компьютерных операционных систем, которые поддерживают TCP/IP, позволяют пользователю статически сконфигурировать IP-адрес на каждом интерфейсе. Маршрутизаторы и коммутаторы также используют статически сконфигурированные IP-адреса.

Серверы также обычно используют статически заданные IP-адреса. Использование статически сконфигурированного и редко изменяемого IP-адреса удобно потому, что все ссылки на этот сервер могут долгое время оставаться неизменными. Очень удобно, если расположение любимого гастронома не меняется и мы всегда знаем, где можно купить продукты и что можно доставить их домой по дороге с работы. Аналогичная концепция действует и для IP-адресов. Если серверы имеют статические, не изменяемые IP-адреса, то пользователи этих серверов знают, как единообразно подключаться к этим серверам из любой точки.

В то же время среднему пользовательскому компьютеру не нужно использовать один и тот же IP-адрес каждый день. Продолжая аналогию с гастрономом, можно сказать, что покупатели могут каждую неделю переезжать на новую квартиру и всё равно будут знать, где находится гастроном, в то время как работникам гастронома не нужно знать, где живут покупатели. Точно так же и серверы обычно не заботятся о том, что персональные компьютеры каждый день получают другие 1P-адреса.

Машины конечных пользователей могут динамически получать IP-адреса и даже изменять их со временем, потому что изменение IP-адреса ни на что не влияет. DHCP определяет протоколы, которые используются для того, чтобы выделять компьютерам IP-адреса. DHCP использует сервер, на котором хранится список диапазонов IP-адресов, доступных в каждой подсети. DHCP-клиент может отправить DHCP-серверу сообщение, запрашивая выделение или аренду IP-адреса. В ответ сервер предлагает IP-адрес. Если этот адрес приемлемый, сервер отмечает, что данный адрес более не доступен для назначения другим узлам, а клиент получает в использование IP-адрес.

Протокол DHCP предоставляет клиентам IP-адреса, а также другую ин формацию. Например, узел должен знать свой IP-адрес, используемую маску подсети, стандартный шлюз, а также IP-адрес (или адреса) DNS-серверов. В большинстве современных сетей DHCP предоставляет конечным узлам всю эту информацию.

На рис. 4 показан типичный набор из четырех сообщений, которые используются между DHCP-сервером и клиентом для назначения IP-адреса и передачи другой информации. Следует заметить, что два первых сообщения являются широковещательными:

1. DHCP-сообщение Discover (широковещательное);
2. DHCP-сообщение Offer для клиента;
3. DHCP-сообщение Requestк серверу;
4. DHCP-сообщение Acknowledgement для клиента.

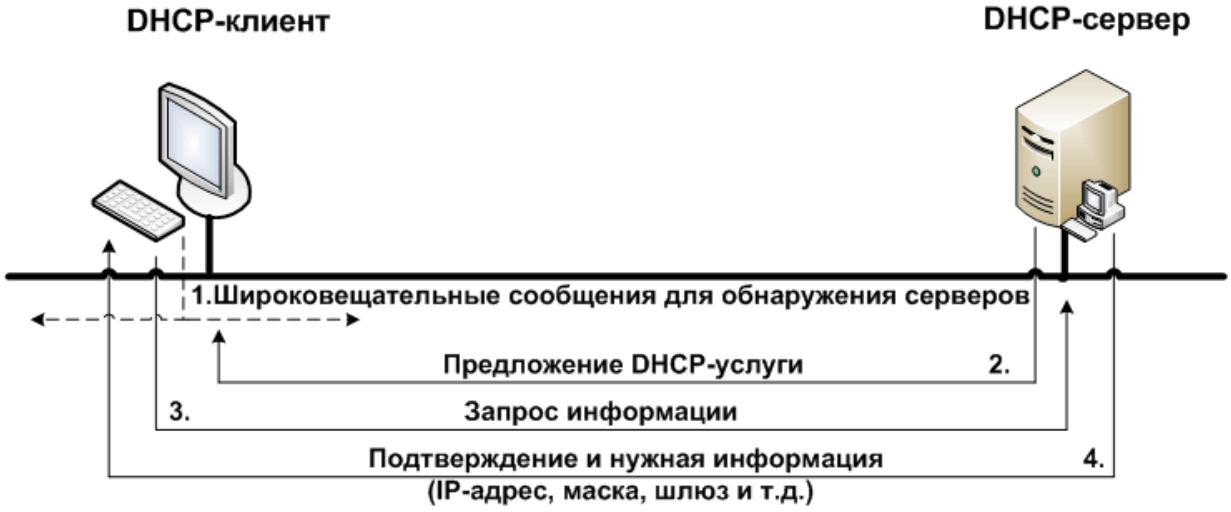


Рис.4 DHCP-сообщения для получения IP-адреса

На рис. 4 DHCP-сервер показан как сервер, что является обычным в сети предприятия. Однако, маршрутизаторы также могут обеспечивать DHCP-службу и делают это. На самом деле маршрутизаторы могут обеспечивать функции DHCP-сервера, динамически назначая IP-адреса компьютерам в небольшой домашней или офисной сети, а так же использовать функции DHCP -клиентов для динамического получения IP-адресов от провайдера интернет-услуг (ISP). Однако необходимость использования этих функций тесно связана с подключением к Интернет.

Протокол DHCP стал весьма распространенным. Большинство узлов конечных пользователей локальных корпоративных сетей получает свои IP-адреса и другую информацию через DHCP.

Протокол ICMP эхо запросов и команда ping

После того как сеть установлена, необходимо проверить базовую IP-связь, не полагаясь на какие-либо приложения. Основным средством для проверки базовой сетевой связи является команда ping. Утилита ping (Packet Internet Groper), использует протокол управляющих сообщений Интернет (Internet Control Message Protocol — ICMP). Она отправляет на другой IP-адрес сообщение, которое называется *эхо-запрос* ICMP (ICMP *echo request*). Ожидается, что компьютер с этим IP- адресом пришлет *эхо-ответ* ICMP (ICMP *echo reply*). Если это так, то сеть успешно протестирована. Иными словами, после этого можно утверждать, что сеть может доставить пакет от одного узла к другому и обратно. Протокол ICMP не полагается на какие-либо приложения, он лишь тестирует базовую IP-связь - уровни 1, 2 и 3 эталонной модели OSI. Этот процесс показан на рис. 5.

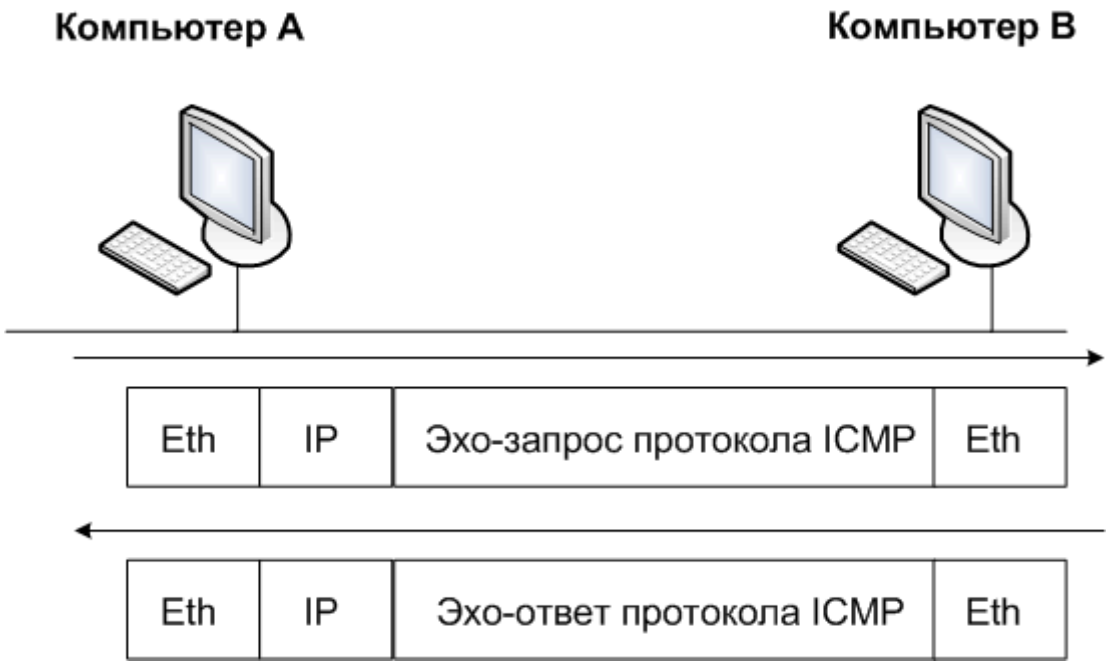


Рис.5 Команда ping в сети

IP-адресация

В этом подразделе рассмотрены некоторые базовые характеристики и особенности IP-адресов. Что наиболее важно, ниже также даются полезные советы о том, как применить имеющиеся знания для практической работы.

Не используйте зарезервированные IP-адреса

Первое, что нужно проверить, относятся ли какие-либо из указанных IP-адресов к зарезервированным (см. Приложение 1) и могут ли они быть использованы для адресации узлов. Все зарезервированные адреса можно разделить на три большие группы:

- адреса, которые всегда зарезервированы (глобально);
- адреса, являющиеся зарезервированными для определенной подсети;
- входят ли адреса в две специализированные подсети в каждой из классовых сетей, а именно, в нулевую и широковещательную подсети.

К первой группе адресов относятся две сети класса A, которые зарезервированы "навсегда" и никогда не будут использоваться, адреса класса D (многоадресатные), а

также IP-адреса класса E (экспериментальные). Такие адреса можно достаточно легко распознать по значению в первом октете:

- 0 (сеть 0.0.0.0 глобально зарезервирована);
- 127 (сеть 127.0.0.0 глобально зарезервирована);
- 224-239 (все сети класса D, многоадресатные);
- 240-255 (все сети класса E, экспериментальные адреса).

Вторая группа, или категория, зарезервированных адресов включает в себя два зарезервированных адреса в каждой из подсетей. Разбивая сеть на подсети, мы не должны использовать для адресации узлов два значения - наименьший и наибольший адреса, известные как:

- адрес (номер) подсети;
- широковещательный адрес.

Поэтому умение быстро и безошибочно определять адрес подсети и широковещательный адрес понадобится, чтобы быстро выбрать из указанного набора адресов не подходящие для присвоения какому-либо узлу.

Третья группа зарезервированных IP-адресов может быть как применима, так и неприменима для какого-либо конкретного вопроса. Для классовой сети, в зависимости от нескольких факторов, следующие две подсети, возможно, придется отбросить при расчетах:

- нулевая подсеть (zero subnet);
- широковещательная подсеть (broadcast subnet).

Одна подсеть с одинаковой маской для каждой сети!

Узлы в одной локальной сети (или в виртуальной сети VLAN), т.е. устройства, относящиеся к одному широковещательному домену, должны относиться к одной и той же адресной подсети. Следовательно, как IP-адрес интерфейса маршрутизатора, так и административный (management) адрес коммутатора и адреса на интерфейсах узлов должны сопровождаться одинаковой маской.

Сетевые настройки оконечного узла

Выше был описан простой алгоритм, используемый узлом при отправке пакета, когда используются протоколы DHCP, DNS, ARP и ICMP. Сам алгоритм можно свести к следующему краткому списку используемых при пересылке данных механизмов.

- **Маршрутизация (routing).** Если IP-адрес получателя пакета находится в той же самой подсети, пакет передается напрямую получателю; если же нет, то пакет пересылается стандартному шлюзу (default gateway).
- **Присвоение адреса (address assignment).** До того как устройство сможет отправить хотя бы один пакет, оно может использовать службы DHCP-клиента для получения своего IP-адреса, маски, стандартного шлюза и адреса DNS-сервера. Перечисленные параметры также могут быть сконфигурированы вручную (т.е. заданы статически) в настройках сети узла.
- **Разрешение имен (паше resolution).** Когда пользователь прямо или косвенно использует имя узла в каком-либо приложении, рабочая станция отправляет DNS-серверу запрос на разрешение имени, чтобы установить IP-адрес нужного узла. Запрос может и не отправляться, если нужная информация уже есть в кэше.
- **Преобразование IP- в MAC-адрес (IP-to-MAC resolution).** Узел использует протокол ARP, чтобы узнать MAC-адрес устройства-получателя или стандартного шлюза. Запрос может и не отправляться, если нужная информация уже есть в кэше.

Из четырех перечисленных выше механизмов, только маршрутизация выполняется для каждого пакета. Функция присвоения адреса обычно используется один раз сразу же после загрузки системы. Процесс разрешения имен и ARP-запросы выполняются по мере необходимости, обычно в ответ на какие-либо действия пользователя.

Чтобы проанализировать, насколько правильно и безошибочно узел использует перечисленные методы, найти и устранить неисправности, нужно знать несколько команд, связанных с проверкой работы сети для оконечной рабочей станции (т.е. узла). В таблице 1 перечислены наиболее важные сетевые команды для операционной системы Windows компании Microsoft, тем не менее, в других операционных системах существуют аналогичные программы или утилиты. В примере 1 ниже проиллюстрирована работа некоторых из перечисленных команд.

Таблица 1

Команды проверки сети операционной системы Windows

Команда	Функции команды
ipconflg /all	Показывает подробную информацию об IP-конфигурации узла, для всех его интерфейсов, в том числе IP-адрес, маску подсети, адреса стандартного шлюза и DNS-сервера
ipconflg /release	Сбрасывает адрес полученный по протоколу DHCP
ipconfig /renew	Запускает процедуру получения IP-адреса и всей связанной с ним информации от DHCP-сервера
nslookup *название*	Отправляет DNS-запрос для указанного *названия* узла
arp -a	Выводит ARP-кэш узла
ipconfig /dieplaydna	Выводит кэш имен узла
ipconfig /fluehdne	Удаляет все записи из кэша имен узла
arp -d	Сбрасывает (очищает) ARP-кэш узла
netstat -rn	Показывает таблицу маршрутизации узла

В примере 1 показана команда ping www. cisco. com, выполненная на узле в операционной системе Windows XP, сразу же после того, как ARP-кэш и кэш имен был удален (т.е. очищен). В первой части примера показаны полученные от DHCP- сервера IP-адрес и DNS-сервер, потом проиллюстрирован процесс очистки кэшей и после этого - команда ping www.cisco.com. Выполнение последнего действия приводит к тому, что узел использует DNS-сервер для получения IP-адреса веб-сервера компании Cisco, после этого выполняет ARP-запрос для обнаружения MAC-адреса стандартного шлюза и только потом пересылает эхо-запрос протокола ICMP по найденному адресу.

*Обратите внимание:*команда ping возвращает отрицательный результат в рассматриваемом примере. Такая ситуация вызвана, скорее всего, списками ACL в маршрутизаторах или брандмауэрах сети Интернет. Тем не менее команда все же запускает процесс генерирования запросов DNS и ARP, как показано в примере 1. Для выполнения команды используется окно командной строки операционной системы Windows XP.

Пример 1. Использование команд для проверки сети в операционной системе рабочей станции

C:\>**ipconfig /all**

! Часть строк опущена.

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . : cinci.rr.com

Description . . . . . : Broadcom NetXtreme 57xx Gigabit Controller

Physical Address . . . . . : 00-11-11-96-B5-13

Dhcp Enabled . . . . . : Yes

Autoconfiguration Enabled . . . . : Yes

IP Address . . . . . : 192.168.1.102

Subnet Mask . . . . . : 255.255.255.0

Default Gateway . . . . . : 192.168.1.1

DHCP Server . . . . . : 192.168.1.1

DNS Servers . . . . . : 65.24.7.3

65.24.7.6

Lease Obtained . . . . . : Thursday, March 29, 2007 6:32:59 AM

Lease Expires . . . . . : Friday, March 30, 2007 6:32:59 AM

! Ниже удаляется кэш ARP и имен.

C:\>**arp -d**

C:\>**ipconfig /flushdns**

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.

! В выводе команды **ping** указан IP-адрес (198.133.219.25),

! что подтверждает работу DNS-сервера.

! Тем не менее команда дает отрицательный результат, возможно из-за

! списков ACL, фильтрующих ICMP-трафик.

C:\>**ping www.cieco.com**

Pinging www.cisco.com [198.133.219.25] with 32 bytes of data:

Request timed out.

Request timed out.

Request timed out.

Request timed out.

Ping statistics for 192.133.219.25:

Packets: Sent - 4, Received = 0, Lost = 4 (100% loss),

! Ниже мы проверяем ARP-кэш и видим запись для стандартного шлюза.

C:\>**arp -a**

Interface: 192.168.1.102 --- 0x2

Internet Address Physical Address Type

192.168.1.1 00-13-10-d4-de-08 dynamic

! Теперь мы проверим кэш имен и убедимся в том, что команда **ping**

! вызвала обращение к DNS-серверу, и имя удаленного узла было получено.

C:\>**ipconfig /displaydns**

Windows IP Configuration

www.cisco.com

Record Name : www.cisco.com

Record Type : 1

Time To Live : 26190

Data Length : 4

Section : Answer

A (Host) Record : 198.133.219.25

! Некоторые строки опущены для краткости

На рис. 6 показан пример конфигурирования IP-адреса узла, маски, стандартного шлюза (default gateway) и DNS-сервера вручную (т.е. статическая конфигурация). Те же самые настройки можно ввести с помощью команд, но большинство пользователей предпочитают использовать графический интерфейс.

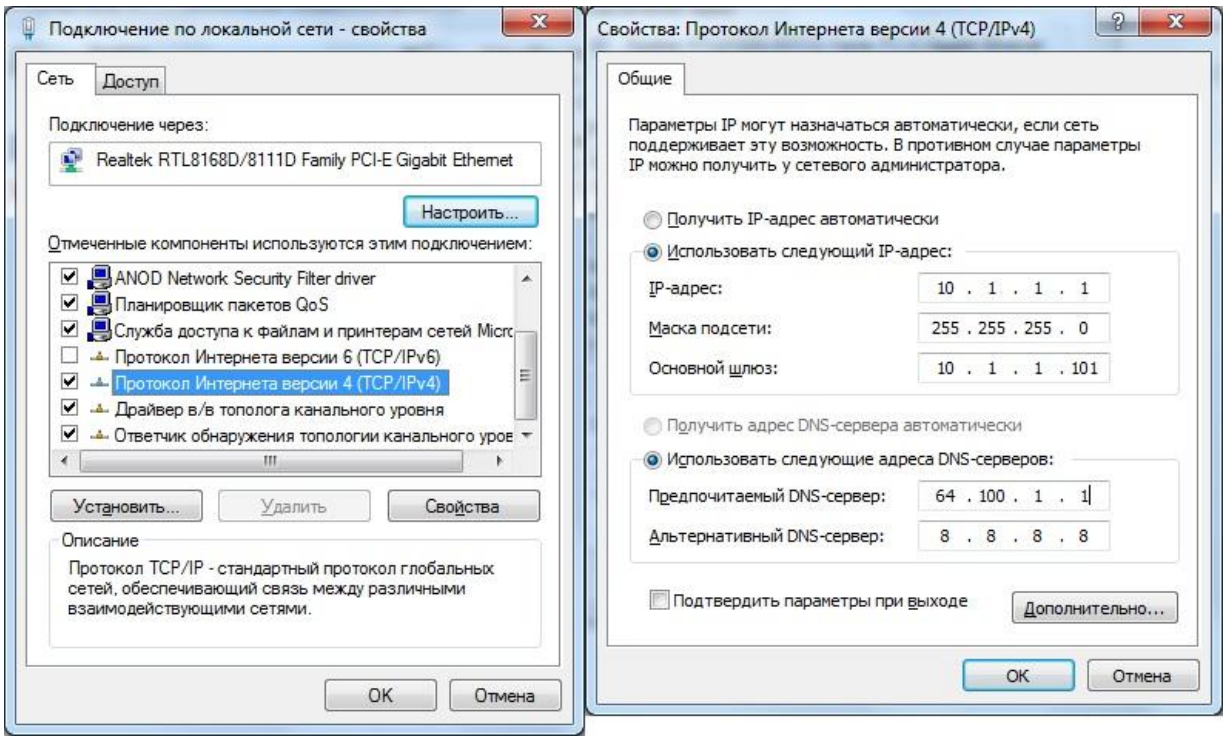


Рис.6 Конфигурирование IP-адреса и параметров сети операционной системы Windows

Поиск и устранение неисправностей в службах маршрутизации узла

Поиск и устранение неисправностей в рассматриваемом случае должны следовать алгоритму работы сети в рабочей станции. Первый вопрос, требующий ответа: «Может ли узел успешно отправлять и принимать ping-пакеты к узлам в той же самой подсети?» Если такая команда ping дает отрицательный результат, то причины неработоспособности могут быть две:

- у двух рабочих станций неправильно сконфигурированы IP-адреса и маски, поэтому один из двух узлов полагает, что относится к другой (тема подсети;
- у двух рабочих станций правильно сконфигурированы IP-адреса и маски, но на уровне Ethernet-среды есть какие-то неисправности.

Если подсети для всех узлов в сети совпадают, тогда следует искать неисправности на уровне 1 и 2 в Ethernet-среде.

Если рабочая станция успешно отправляет и принимает ping-пакеты от узла в той же самой подсети, следует проверить, проходят ли запросы к IP-адресам из других подсетей и таким образом проверить следующий этап алгоритма маршрутизации пакетов узлом. Две проверки пригодятся следующие действия:

- выполните ping-запросы к IP-адресу стандартного шлюза, чтобы убедиться, что узел может пересылать пакеты через локальную сеть к стандартному шлюзу и получать ответы;
- выполните ping-запросы к IP-адресу из другой посети со стандартного шлюза (маршрутизатора), но не к адресу узла в локальной сети.

Например, как показано в примере 1, на рабочей станции ПК1 можно выполнить команду ping 172.16.1.253, чтобы проверить, что она может отправлять и принимать пакеты от стандартного шлюза. Если команда даст положительный результат, на узле ПК1 можно выполнить команду ping 172.16.2.253, благодаря которой рабочая станция будет использовать настройки стандартного шлюза, поскольку для ПК1 адрес 172.16.2.253 находится в другой подсети.

Итак, если тестирование с помощью программы ping дает положительный результат для узлов в той же самой подсети и отрицательный - для узлов в других подсетях, то типичные причины такой ошибки могут быть следующими.

- Есть несоответствие между настройками стандартного шлюза узла и конфигурацией самого стандартного шлюза или маршрутизатора, выступающего в качестве такового. Проблема может заключаться в несоответствии масок узла и маршрутизатора, что, опять же, повлияет на то, как устройства идентифицируют диапазоны адресов подсети, или узел может считать неправильный IP-адрес стандартным шлюзом.
- Если настройки стандартного шлюза правильны, но ping-тестирование шлюза дает отрицательный результат, то в локальной сети есть проблема на первом или втором уровнях (Layer 1 или 2).
- Если настройки стандартного шлюза правильны и ping-тестирование шлюза дает положительный результат, но ping-тестирование одного из интерфейсов маршрутизатора дает отрицательный результат (например, ping 172.16.2.253, см. Пример1), то это может свидетельствовать о том, что отказал интерфейс маршрута затора.

В этом разделе описываются различные сетевые проблемы и методы их обнаружения для рабочих станций, тем не менее, следует помнить, что самая основная и часто встречающаяся ошибка - несоответствие IP-адресов и масок узлов в сети.

Сбор сетевых данных с помощью программы Wireshark

Wireshark, широко известный анализатор сетевых протоколов и средство мониторинга. Программа Wireshark собирает все пакеты, отправленные или полученные сетевой интерфейсной платой (NIC) компьютера. Ее можно установить либо в лаборатории, либо дома на ПК. Вам он понадобиться для отслеживания и просмотра разных типов сетевых протоколов и трафика. Ранее программа Wireshark была известна под именем Ethereal.

Программа Wireshark поставляется бесплатно и доступна по адресу www.wireshark.org.

Установка и запуск программы Wireshark

Если программа Wireshark загружалась на ПК ранее, перейдите в папку с программой Wireshark Start > All Programs > Wireshark > Wireshark (пуск > программы > Wireshark > Wireshark) и щелкните значок приложения.

Если ранее программа Wireshark не устанавливалась, выполните следующие действия:

а. Указав путь в локальной сети до установщика программы Wireshark, wireshark-setup-0.99.5.exe, загрузите установщик на рабочий стол ПК.

б. Щелкните установщик два раза и следуйте его подсказкам, принимая значения по умолчанию (рис.7).

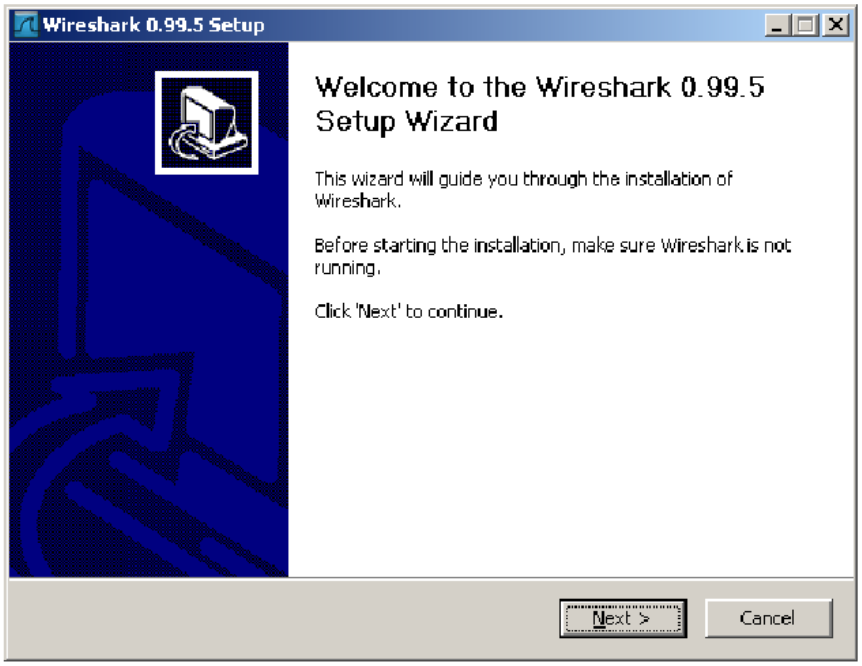


Рис.7 Установка программы Wireshark

1) Нажмите кнопку I Agree (принять), см. рис.8.

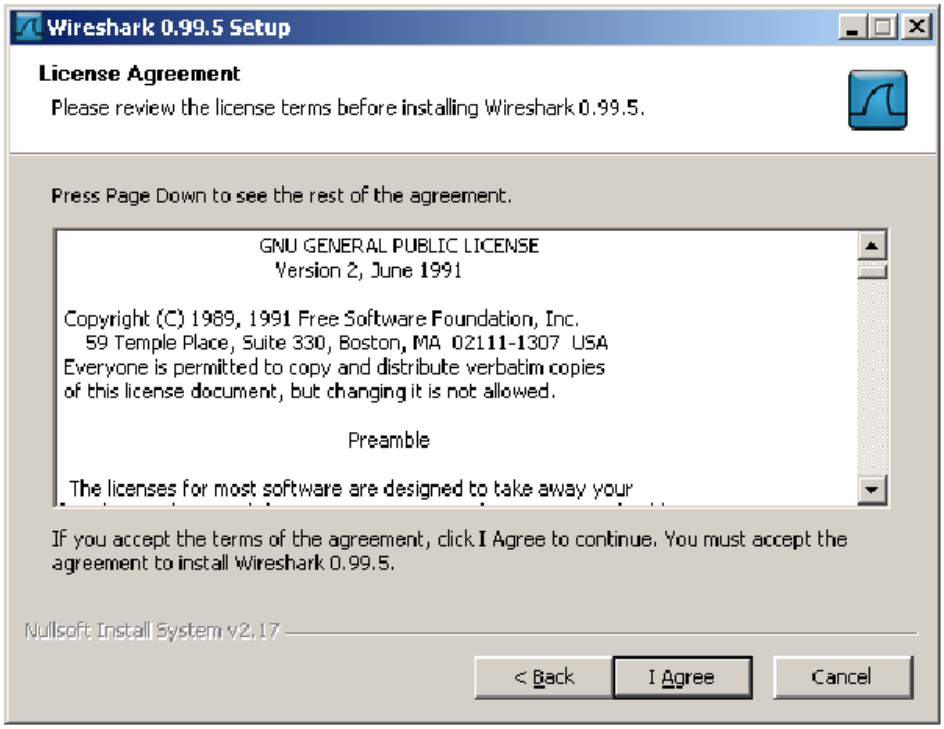


Рис.8 Лецензионное соглашение

2) Убедитесь, что на ПК установлен WinPcap. В WinPcap входит драйвер для поддержки сбора пакетов. Программа Wireshark использует эту библиотеку для сбора динамических сетевых данных в Windows.

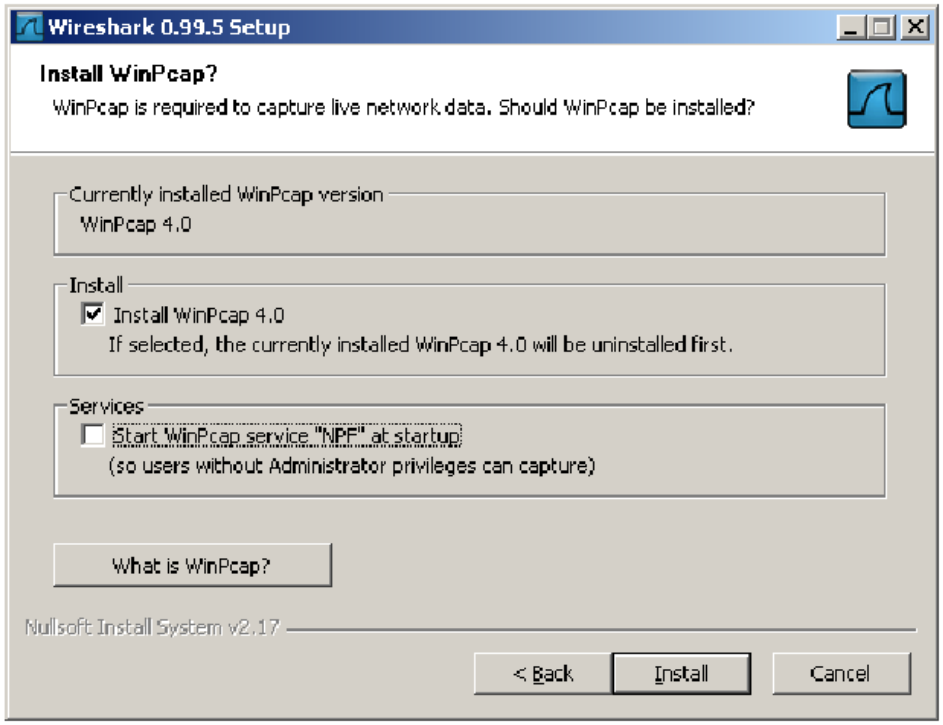


Рис.9 Окно установки WinPcap

в. Щелкните Install (установить) и следуйте подсказкам до конца процесса установки.

г. После установки программы установите соответствующий флажок, чтобы

запустить программу Wireshark.

Выбор интерфейса для сбора пакетов

а. Запустите приложение Wireshark.

б. В меню Capture (сбор) выберите пункт Interfaces (интерфейсы), см. рис.10.

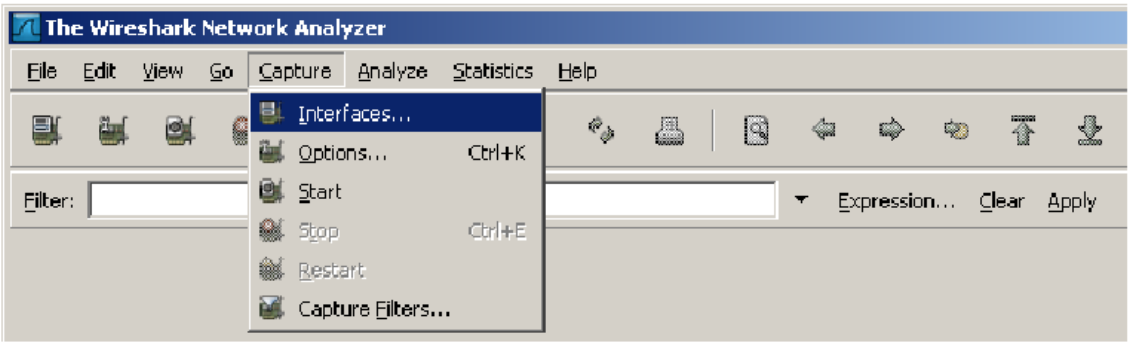


Рис.10 Интерфейс программы

в. Нажмите кнопку Start (пуск) для интерфейса Ethernet (NIC), который требуется использовать для сбора сетевого трафика (рис.11).

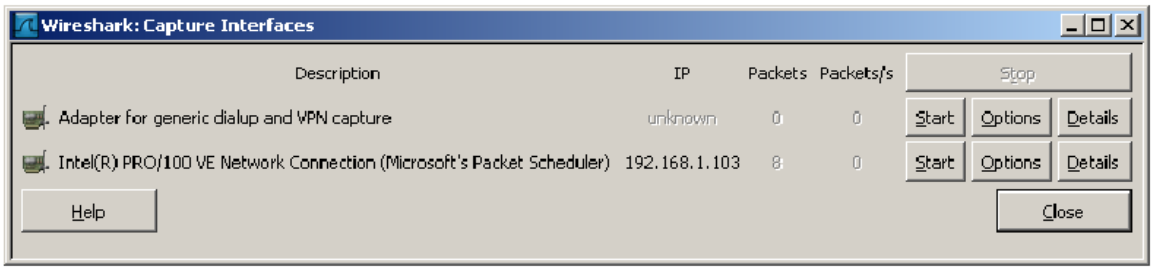


Рис.11 Выбор интерфейса для сбора статистики

Запуск сбора сетевых данных

а. Прокрутите меню и просмотрите панель инструментов в интерфейсе запуска Wireshark.

б. Нажмите кнопку New Live Capture (новый сбор динамических данных) и просмотрите сведения, собранные Wireshark (рис.12). Пусть сбор данных продолжается в течение нескольких минут, чтобы вы могли понаблюдать за различными типами трафика в сети .

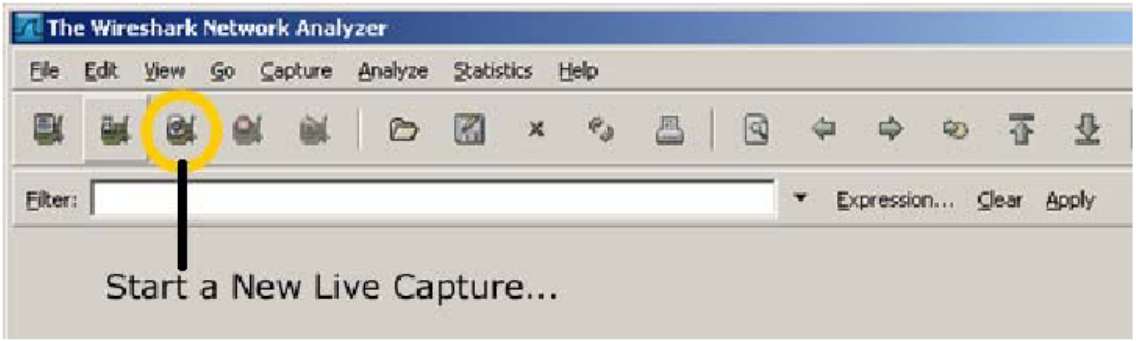


Рис.12 Процесс запуска сбора сетевых данных

Анализ сведений о веб-трафике

a. Если существует подключение к сети Интернет, откройте веб-обозреватель и перейдите в узел www.google.com. Сверните окно Google и вернитесь в Wireshark. Должен быть отображен трафик, схожий с тем, что представлен ниже (рис.13). Найдите столбцы Source, Destination и Protocol (источник, адрес назначения и протокол) на экране Wireshark.

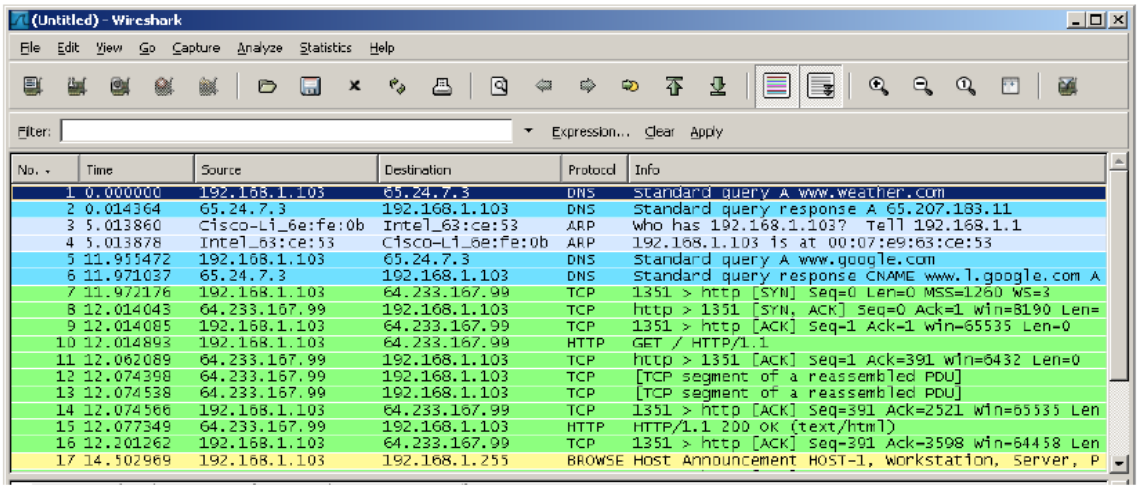


Рис.13 Анализ трафика

Подключение к серверу Google начнется с отправки запроса на DNS-сервер для поиска IP-адреса сервера. IP-адрес сервера назначения, по всей вероятности, начнется с 64.x.x.x. Каковы источник и адрес назначения первого пакета, отправленного на сервер Google?

Фильтрация сбора сетевых данных

a. Откройте окно командной строки, выбрав Start > All Programs > Run (пуск > программы > выполнить) и введя cmd. Либо щелкните Start > All Programs > Accessories

(пуск > все программы > стандартные > командная строка).

б. Отправьте эхо-запрос по IP-адресу узла в вашей локальной сети и понаблюдайте за процессами в окне сбора Wireshark. Прокрутите вниз и вверх окно, в котором отображается трафик. Какие используются типы протоколов?

в. В текстовом поле Filter (фильтр) введите icmp и щелкните Apply (применить). Протокол управления сообщениями в Интернет (ICMP) — это протокол, используемый эхо-запросом для проверки сетевого подключения к другому узлу см. рис.14.

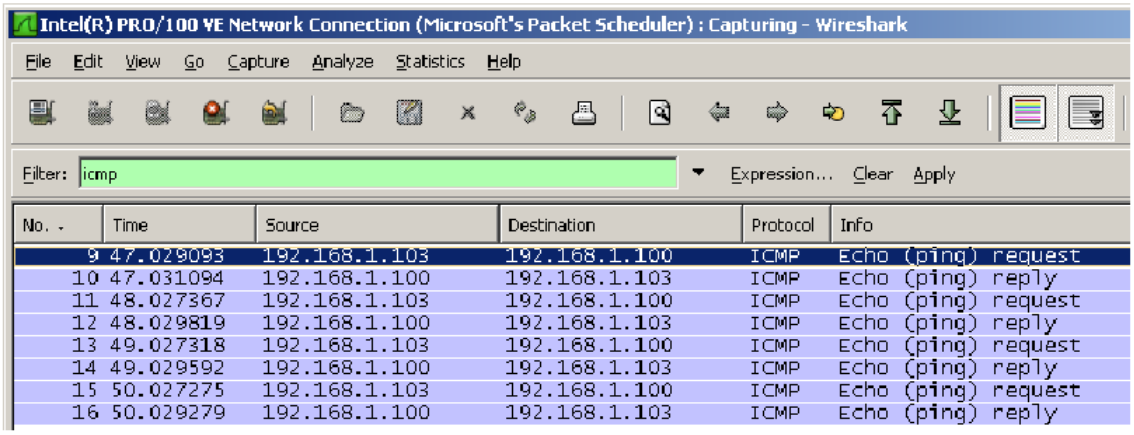


Рис.14 Фильтрация сбора сетевых данных

Задание на лабораторную работу

1. Ознакомиться с теоретической частью используя дополнительную литературу.
2. Исследовать штатные средства диагностики Windows.
3. Выполнить сбор сетевого трафика с помощью программы Wireshark, чтобы ознакомиться с интерфейсом и средой Wireshark;
    a. Проанализировать трафик для веб-сервера;
    b. Создать фильтр для ограничения сбора сетевых данных пакетами ICMP.
    c. Отправить эхо-запрос удаленному узлу, чтобы понаблюдать за работой фильтра пакетов ICMP в ходе сбора сетевых данных.

Требования к содержанию отчета

1. Отчёт должен содержать:
2. Краткие теоретические сведения.
3. Задание.
4. Результаты выполнения задания.
5. Результаты выполнения программного диалога.

Литература

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.
2. Одом, Уэнделл. Официальное руководство Cisco по подготовке к сертифицированным экзаменам CCENT/CCNA ICND1. 2-е изд. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011 – 672с.
3. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.
4. http://www.wireshark.org/docs/ //Документация по программе wireshark

Приложение 1

Классы сетей

Для того, чтобы как-то структурировать сети, их поделили на классы. Классы сетей были введены в 1981 году на заре рождения интернета. Тогда никто еще не беспокоился о возможности исчерпания адресного пространства. Именовали принцип класификации словом **«classful».** Всего использовалось пять классов сетей: **A, B, C, D, E**. Первые три класса для адресации сетей, другие два имели специальное назначение.

В таблице П1 приведены диапазоны номеров сетей, соответствующих каждому классу сетей. Количество бит отведенное в каждом классе для адреса сети и адреса хоста наглядно представлено на рис. П1.

Таблица П1

Диапазоны номеров сетей

Класс	Наименьший адрес	Наибольший адрес
A	1.0.0.0	126.0.0.0
B	128.0.0.0	191.255.0.0
C	192.0.0.0.	223.255.255.0
D	224.0.0.0	239.255.255.255
E	240.0.0.0	247.255.255.255

Рис.П1 Приведены классы сетей

Приложение 2

Штатные средства диагностики сети в Windows

GUI Windows интерпретирует сетевые подключения как объекты, которыми пользователь может управлять и о которых может получать информацию.

В состоянии любого активного подключения можно узнать все параметры конфигурации TCP/IP (IP, Маска подсети, Основной шлюз) см. рис П2.1 .

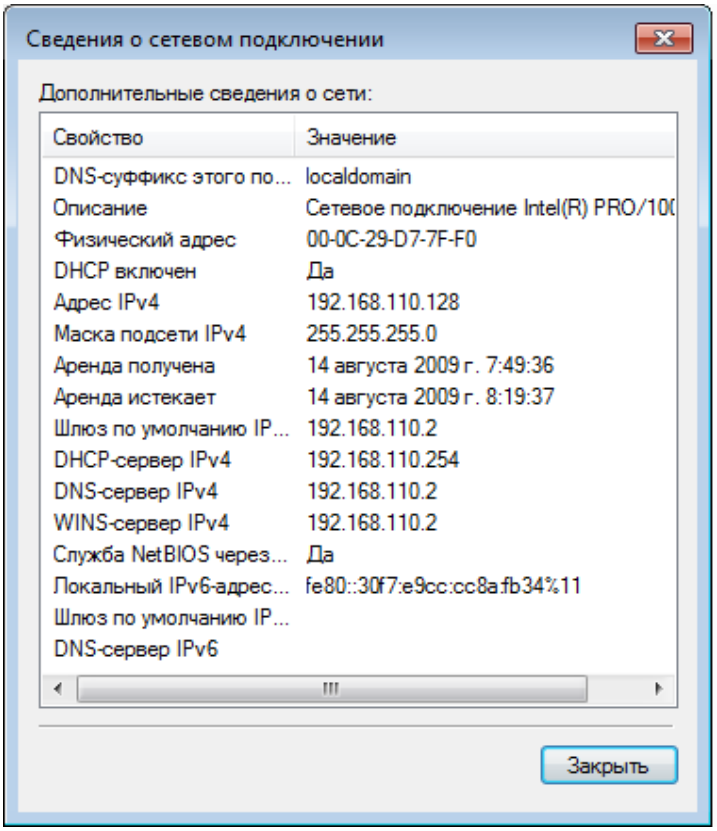


Рис.П2.1 Сведения о сетевом подключении

Сетевые подключения можно найти следующим образом: Пуск → Панель Управления → Сетевые подключения (для Windows XP при просмотре значков Панели управления в Классическом виде) **или** Пуск → Панель Управления → Центр управления сетями и общим доступом → Управление сетевыми подключениями/Изменение параметров адаптера (Для Vista и 7 при просмотре Панели управления в режиме «Мелкие значки»).

В свойствах сетевых подключений можно задавать конфигурацию протоколов вручную (рис.П2.1-П2.2):

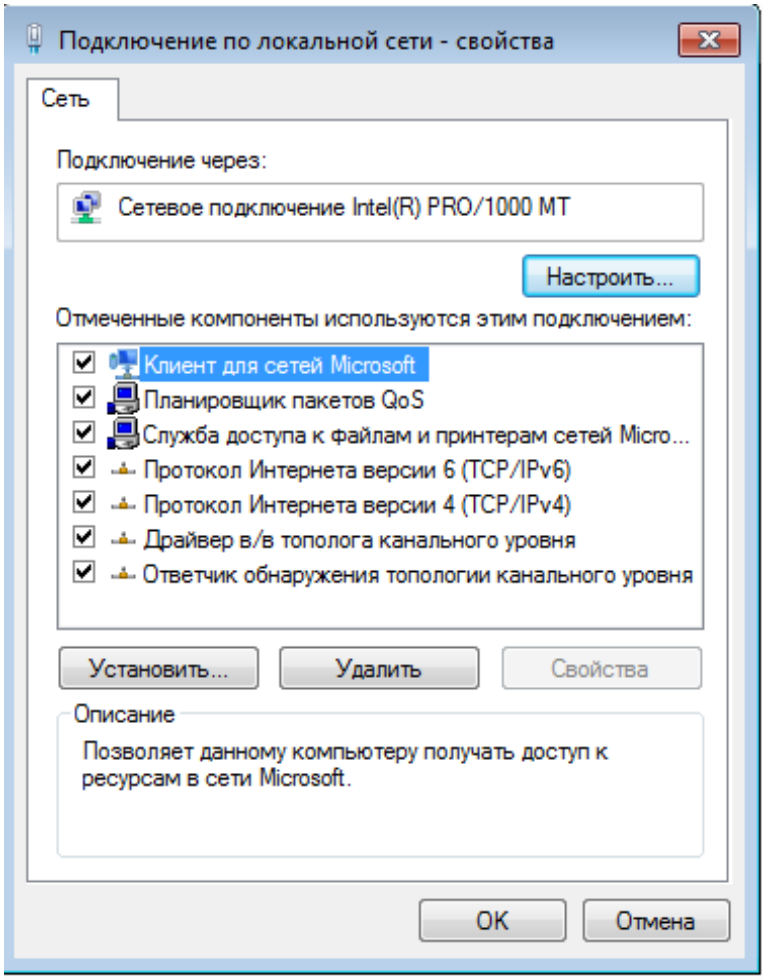


Рис.П2.2 Свойства сетевого соединения

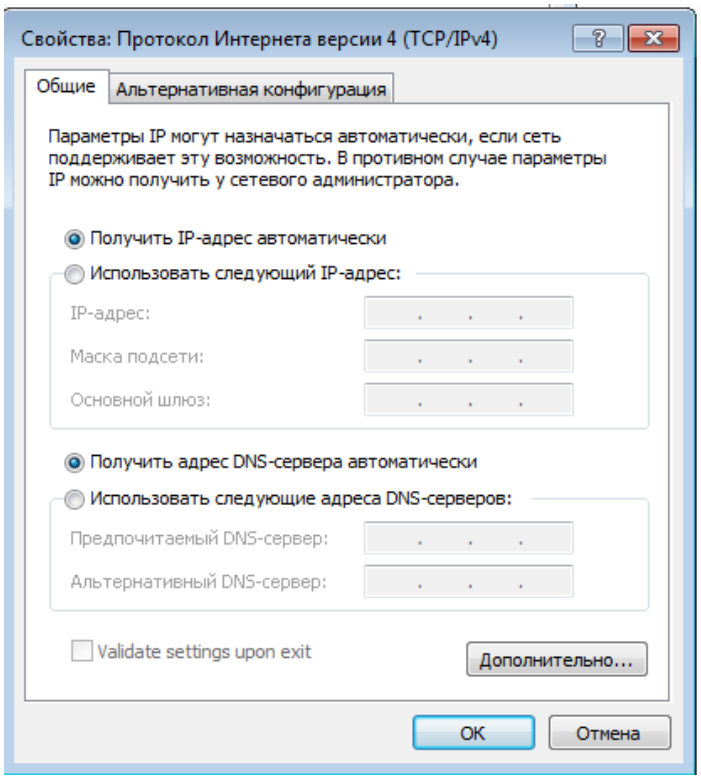


Рис.П2.3 Свойства протокола TCP/IP сетевого соединения

Вывод команд консоли

Пуск, выполнить, cmd

Пуск, все программы, стандартные, командная строка.

ipconfig.exe

Местонахождение: Windows\system32

Описание: Утилита командной строки Ipconfig служит для отображения всех текущих параметров сети TCP/IP и обновления параметров DHCP и DNS (см. рис.П2.4).

При вызове команды ipconfig без параметров выводится только IP-адрес, маска подсети и основной шлюз для каждого сетевого адаптера

Синтаксис: ipconfig [/all] [/renew [адаптер]] [/release [адаптер]] [/flushdns] [/displaydns] [/registerdns] [/showclassid адаптер] [/setclassid адаптер [код_класса]]

```
C:\Documents and Settings\Администратор>ipconfig

Настройка протокола IP для Windows

Подключение по локальной сети - Ethernet адаптер:

        DNS-суффикс этого подключения . . :
        IP-адрес  . . . . . . . . . . . . : 172.25.90.222
        Маска подсети . . . . . . . . . . : 255.255.255.0
        Основной шлюз . . . . . . . . . . : 172.25.90.1

Интернет - PPP адаптер:

        DNS-суффикс этого подключения . . :
        IP-адрес  . . . . . . . . . . . . : 95.154.100.143
        Маска подсети . . . . . . . . . . : 255.255.255.255
        Основной шлюз . . . . . . . . . . : 95.154.100.143
```

Рис.П2.4 Вывод команды ipconfig

nslookup.exe

Местонахождение: Windows\system32

Описание: Предоставляет сведения, предназначенные для диагностики инфраструктуры DNS. Для использования этого средства необходимо быть знакомым с принципами работы системы DNS. Средство командной строки Nslookup доступно (см. рис.П2.5), только если установлен протокол TCP/IP

Синтаксис: nslookup [-подкоманда …] [{искомый_компьютер| [-сервер]}]

```
C:\Documents and Settings\Администратор>nslookup mail.ru
Server:  ns2.inetvl.ru
Address:  95.154.112.74

Non-authoritative answer:
Name:    mail.ru
Addresses:  217.69.128.45, 217.69.128.41, 217.69.128.42, 217.69.128.43
          217.69.128.44
```

Рис. П2.5 Вывод команды nslookup

netstat.exe

Местонахождение: Windows\system32

Описание: Отображение активных подключений TCP, портов, прослушиваемых компьютером, статистики Ethernet, таблицы маршрутизации IP, статистики IPv4 (для протоколов IP, ICMP, TCP и UDP) и IPv6 (для протоколов IPv6, ICMPv6, TCP через IPv6 и UDP через IPv6) (см. рис.П2.6). Запущенная без параметров, команда nbtstat отображает подключения TCP.

Синтаксис: netstat [-a] [-e] [-n] [-o] [-p протокол] [-r] [-s] [интервал]

```
C:\Documents and Settings\Администратор>netstat -abo

Активные подключения

  Имя     Локальный адрес         Внешний адрес          Состояние       PID
  TCP     neiroman:pptp           neiroman:0             LISTENING       4
  [Система]

  TCP     neiroman:50300          neiroman:0             LISTENING       1604
  [oodag.exe]

  TCP     neiroman:50303          neiroman:0             LISTENING       1604
  [oodag.exe]

  TCP     neiroman:30606          neiroman:0             LISTENING       1548
  [ekrn.exe]

  TCP     neiroman:1848           localhost:30606        ESTABLISHED     736
  [chrome.exe]

  TCP     neiroman:1850           localhost:30606        ESTABLISHED     736
  [chrome.exe]

  TCP     neiroman:30606          localhost:1850         ESTABLISHED     1548
  [ekrn.exe]

  TCP     neiroman:30606          localhost:1848         ESTABLISHED     1548
  [ekrn.exe]

  TCP     neiroman:1059           172.16.4.1:pptp        ESTABLISHED     4
  [Система]

  TCP     neiroman:1357           asterix.inetvl.ru:22   ESTABLISHED     580
  [putty.exe]

  TCP     neiroman:1849           whiskey.inetvl.ru:http  ESTABLISHED     1548
  [ekrn.exe]

  TCP     neiroman:1851           stat.inetvl.ru:http    ESTABLISHED     1548
  [ekrn.exe]
```

Рис.П2.6 Вывод команды netstat

ping.exe

Местонахождение: Windows\system32

Описание: Утилита командной строки Ping проверяет соединение на уровне протокола IP с другим компьютером, поддерживающим TCP/IP, с помощью отправки сообщений с эхо-запросом по протоколу ICMP .

После каждой передачи выводится соответствующее сообщение с эхо-ответом (см.рис.П2.7). Ping - это основная TCP/IP-команда, используемая для устранения неполадки в соединении, проверки возможности доступа и разрешения имен. Команда ping, запущенная без параметров, выводит справку.

Синтаксис: ping [-t] [-a] [-n счетчик] [-l размер] [-f] [-i TTL] [-v тип] [-r счетчик] [-s счетчик] [{-j список_узлов | -k список_узлов}] [-w интервал] [имя_конечного_компьютера]

```
C:\Documents and Settings\Администратор>ping -l 1470 -n 20 172.25.90.1

Обмен пакетами с 172.25.90.1 по 1470 байт:

Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=12мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=2мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=9мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=12мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=15мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=14мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=6мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=1мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=3мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=2мс TTL=64
Ответ от 172.25.90.1: число байт=1470 время=6мс TTL=64

Статистика Ping для 172.25.90.1:
    Пакетов: отправлено = 20, получено = 20, потеряно = 0 (0% потерь),
Приблизительное время приема-передачи в мс:
    Минимальное = 1мсек, Максимальное = 15 мсек, Среднее = 4 мсек
```

Рис.П2.7 Вывод команды ping

pathping.exe

Местонахождение: Windows\system32

Описание: Утилита командной строки pathping.exe в течение некоторого периода времени отправляет многочисленные сообщения с эхо-запросом каждому маршрутизатору, находящемуся между исходным пунктом и пунктом назначения, а затем на основании пакетов, полученных от каждого из них, вычисляет результаты (см.рис.П2.8).

Поскольку pathping показывает коэффициент потери пакетов для каждого маршрутизатора или связи, можно определить маршрутизаторы или субсети, имеющие проблемы с сетью. Команда Pathping выполняет эквивалентное команде tracert действие, идентифицируя маршрутизаторы, находящиеся на пути.

Затем она периодически в течение заданного времени обменивается пакетами со всеми маршрутизаторами и на основании числа пакетов, полученных от каждого из них, обрабатывает статистику. Запущенная без параметров, команда pathping выводит справку.

Синтаксис: pathping [-n] [-h максимальное_число_переходов] [-g список_узлов] [-p период] [-q число_запросов [-w интервал] [-T] [-R] [имя_конечного_компьютера]

```
C:\Documents and Settings\Администратор>pathping mail.ru

Трассировка маршрута к mail.ru [217.69.128.45]
с максимальным числом прыжков 30:
  0  neiroman [95.154.100.143]
  1  95.154.112.117
  2  95.154.112.1
  3  inetvl-ptcom-gw.ptkom.ru [85.95.130.170]
  4  vvk15.vvk25.transtelecom.net [217.150.59.38]
  5  TTK-gw.Moscow.gldn.net [194.186.0.194]
  6  TTK-lgw.Moscow.gldn.net [194.186.0.193]
  7  cat01.Moscow.gldn.net [194.186.158.110]
  8     *     mailru-KK12-1-gw.Moscow.gldn.net [195.239.8.10]
  9  reserved.128.69.217.in-addr.arpa [217.69.128.45]

Подсчет статистики за: 225 сек. ...
           Исходный узел     Маршрутный узел
Прыжок   RTT    Утер./Отпр.   %   Утер./Отпр.  %   Адрес
  0                                              neiroman [95.154.100.143]
                                 0/ 100 =  0%   |
  1    0мс      0/ 100 =  0%     0/ 100 =  0%  95.154.112.117
                                 0/ 100 =  0%   |
  2    8мс      0/ 100 =  0%     0/ 100 =  0%  95.154.112.1
                                 0/ 100 =  0%   |
  3   10мс      2/ 100 =  2%     2/ 100 =  2%  inetvl-ptcom-gw.ptkom.ru [85.95.130.170]
                                 0/ 100 =  0%   |
  4   20мс      3/ 100 =  3%     3/ 100 =  3%  vvk15.vvk25.transtelecom.net [217.150.59.38]
                                 0/ 100 =  0%   |
  5  136мс      3/ 100 =  3%     3/ 100 =  3%  TTK-gw.Moscow.gldn.net [194.186.0.194]
                                 0/ 100 =  0%   |
  6  226мс      2/ 100 =  2%     2/ 100 =  2%  TTK-lgw.Moscow.gldn.net [194.186.0.193]
                                 0/ 100 =  0%   |
  7  142мс      0/ 100 =  0%     0/ 100 =  0%  cat01.Moscow.gldn.net [194.186.158.110]
                                 1/ 100 =  1%   |
  8  137мс      1/ 100 =  1%     0/ 100 =  0%  mailru-KK12-1-gw.Moscow.gldn.net [195.239.8.10]
                                 0/ 100 =  0%   |
  9  137мс      1/ 100 =  1%     0/ 100 =  0%  reserved.128.69.217.in-addr.arpa [217.69.128.45]

Трассировка завершена.
```

Рис.П2.8 Вывод команды pathping

route.exe

Местонахождение: Windows\system32

Описание: Эта команда нужна для редактирования или просмотра таблицы маршрутов IP из командной строки. Ключ /? выводит все доступные ключи при работе с Route см.рис.П2.9.

```
C:\Documents and Settings\Администратор>route print
===========================================================================
Список интерфейсов
0x1 ........................... MS TCP Loopback interface
0x2 ...00 1b fc 32 6f 5b ...... Realtek RTL8168/8111 PCI-E Gigabit Ethernet NIC
0x20004 ...00 53 45 00 00 00 ...... WAN (PPP/SLIP) Interface
===========================================================================
===========================================================================
Активные маршруты:
Сетевой адрес           Маска сети      Адрес шлюза       Интерфейс  Метрика
          0.0.0.0          0.0.0.0   95.154.100.143  95.154.100.143       1
          0.0.0.0          0.0.0.0      172.25.90.1   172.25.90.222      21
   95.154.100.143  255.255.255.255        127.0.0.1       127.0.0.1      50
    95.154.112.64  255.255.255.192      172.25.90.1   172.25.90.222       1
     95.154.113.0  255.255.255.128      172.25.90.1   172.25.90.222       1
   95.255.255.255  255.255.255.255   95.154.100.143  95.154.100.143      50
        127.0.0.0        255.0.0.0        127.0.0.1       127.0.0.1       1
       172.16.0.0      255.240.0.0      172.25.90.1   172.25.90.222       1
       172.16.4.1  255.255.255.255      172.25.90.1   172.25.90.222       1
      172.25.90.0    255.255.255.0    172.25.90.222   172.25.90.222      20
    172.25.90.222  255.255.255.255        127.0.0.1       127.0.0.1      20
   172.25.255.255  255.255.255.255    172.25.90.222   172.25.90.222      20
      192.168.0.0      255.255.0.0      172.25.90.1   172.25.90.222       1
        224.0.0.0        240.0.0.0    172.25.90.222   172.25.90.222      20
        224.0.0.0        240.0.0.0   95.154.100.143  95.154.100.143       1
  255.255.255.255  255.255.255.255   95.154.100.143  95.154.100.143       1
  255.255.255.255  255.255.255.255    172.25.90.222   172.25.90.222       1
Основной шлюз:      95.154.100.143
===========================================================================
Постоянные маршруты:
  Сетевой адрес           Маска    Адрес шлюза      Метрика
       172.16.0.0      255.240.0.0     172.25.90.1       1
      192.168.0.0      255.255.0.0     172.25.90.1       1
     95.154.113.0  255.255.255.128     172.25.90.1       1
    95.154.112.64  255.255.255.192     172.25.90.1       1
```

Рис.П2.9 Вывод команды Route

tracert.exe

Местонахождение: Windows\system32

Описание: Определяет путь до точки назначения с помощью посылки в точку назначения эхо-сообщений протокола Control Message Protocol (ICMP) с постоянным увеличением значений срока жизни (Time to Live, TTL).

Выведенный путь - это список ближайших интерфейсов маршрутизаторов, находящихся на пути между узлом источника и точкой назначения. Ближний интерфейс представляют собой интерфейс маршрутизатора, который является ближайшим к узлу отправителя на пути, см.рис.П2.10.

Например, чтобы вывести трассу маршрута к http://www.microsoft.com, нужно набрать: C:\>tracert www.microsoft.com

Синтаксис: tracert [-d] [-h максимальное_число_переходов] [-j список_узлов] [-w интервал] [имя_конечного_компьютера]

```
C:\Documents and Settings\Администратор>tracert vl.ru

Трассировка маршрута к vl.ru [80.92.162.132]
с максимальным числом прыжков 30:

  1     3 ms     2 ms     3 ms  95.154.112.117
  2     5 ms     5 ms     4 ms  95.154.112.1
  3     5 ms     4 ms     6 ms  109.126.0.129
  4     4 ms     5 ms     4 ms  109.126.0.165
  5     4 ms     5 ms     5 ms  109.126.0.150
  6     7 ms     5 ms     4 ms  vl.ru [80.92.162.132]

Трассировка завершена.
```

Рис.П2.10 Вывод команды tracert

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК
Кафедра информационных систем управления

С. С. Пашин

СЖАТИЕ ДАННЫХ

Методические указания к лабораторной работе
по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток
2013

Приведены основные сведения о методах сжатия данных, подробно рассмотрен алгоритм LZ78. Цель указаний – формирование у студентов понятия об проблеме избыточности представления на физических носителях и в системах передачи данных информации, и ознакомления с алгоритмами позволяющими решить в какой т остепени эту проблему

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

СЖАТИЕ ДАННЫХ

Одной из проблем систем хранения и передачи информации является проблема ее избыточного представления на физических носителях и в системах передачи данных. На решение выше описанной проблемы направлена технология сжатия данных, включающая в себя множество методов и алгоритмов упаковки (устранения избыточности) информации.

История развития теории сжатия информации

В сороковых годах ученые, работающие в области информационных технологий, ясно поняли, что можно разработать такой способ хранения данных, при котором пространство будет расходоваться более экономно. Клод Шеннон, изучая нюансы различий между семантикой (semantics) (что некая сущность значит) и синтаксисом (syntax) (как некая сущность выражается), разработал большинство базовых понятий этой теории. Понимание того, что одно и то же значение (семантика) может быть реализовано различными способами (синтаксис), приводит к закономерному вопросу: "Какой способ выражения чего-либо является наиболее экономичным?" Поиск ответа на этот вопрос привел Шеннона к мысли об энтропии, которая, проще говоря, соотносится с количеством, содержащейся в файле полезной информации. Методы сжатия пытаются увеличивать энтропию файла, то есть уменьшать длину файла, сохраняя при этом всю информацию.

Однако, Шеннон не был первым, кто задумывался о сущности информации и определении ее количества. Первый шаг на этом пути сделал в 1928 г. Хартли. Основной полученный им результат можно сформулировать примерно так: если в заданном множестве, содержащем N элементов, выделен некоторый элемент x, о котором известно лишь, что он принадлежит этому множеству, то, чтобы найти x, необходимо получить количество информации, равное log2 N. Эту формулу обычно называют формулой Хартли.

Формула Хартли является частным случаем более общей формулы Шеннона, позволяющей найти количество информации в случайном сообщении фиксированного алфавита. Пусть X1, ..., Xn - символы этого алфавита, P1, ..., Pn - вероятности их появления в тексте сообщения, тогда формула Шеннона принимает вид:

$$H = P1*\log2(1 / P1) + ... + Pn*\log2(1 / Pn),$$

где H - количество бит информации в одном символе сообщения, или энтропия символа сообщения. Это число показывает минимальное среднее число бит, необходимых для представления одного символа алфавита данного сообщения.

В некоторых случаях алфавит сообщения может быть неизвестен, тогда выдвигаются гипотезы об алфавите сообщения. Имея разные алфавиты, можно достичь разных коэффициентов сжатия. Например, текстовый файл, если его рассматривать как последовательность битов, имеет энтропию порядка 0.7 - 0.9, если как последовательность байтов, - 0.5 - 0.7, хотя популярные программы сжатия уменьшают размеры текстовых файлов до 0.3 - 0.4 от исходного размера.

Доказательство Шеннона не было конструктивным, т.е. не содержало способа построения этих оптимальных кодов, а лишь показывало их существование. До появления работы Шеннона, кодирование символов алфавита при передаче сообщения по каналам связи осуществлялось одинаковым количеством бит, получаемым по формуле Хартли. С появлением этой работы начали появляться способы, кодирующие символы разным числом бит в зависимости от вероятности появления их в тексте. Например, часто в файлах некоторые значения байта встречаются чаще других. Таким образом, за счет использования для каждого значения байта кода различной длины можно значительно уменьшить общий размер данных. Эта базовая идея лежит в основе алгоритмов сжатия Шеннона-Фано (Shannon-Fano) и Хаффмана (Huffman). Подобные алгоритмы выбирают более короткие коды для часто встречающихся и более длинные для редко встречающихся значений байта. Обычно текстовые файлы (в которых одни значения байтов повторяются гораздо чаще других) они сжимают довольно хорошо.

Более тридцати лет алгоритм сжатия Хаффмана и его варианты оставались наиболее популярными методами. Однако в 1977 два исследователя из Израиля предложили совершенно другой подход к этой проблеме. Абрахам Лемпел и Якоб Зив выдвинули идею формирования "словаря" общих последовательностей данных. При этом сжатие данных осуществляется за счет замены записей соответствующими кодами из словаря. Существуют два алгоритма, в настоящее время известные как LZ77 и LZ78. Они уже не требуют включения словаря данных в архив, так как если вы формируете ваш словарь определенным способом, программа декодирования может его восстанавливать непосредственно из ваших данных. К сожалению, LZ77 и LZ78 тратят много времени на создание эффективного словаря. Лемпел был приглашен фирмой Sperry для оказания им

помощи в разработке способа наиболее эффективной упаковки данных на компьютерных лентах. В этой же фирме Терри Велч (Terry Welch) расширил алгоритм LZ78, создав новый вариант, широко известный, как LZW.

На работу Велча обратила внимание группа программистов Unix и использовала его алгоритм в их приложении LZW, получившем вполне естественное название compress. Они добавили несколько усовершенствований и опубликовали общедоступную версию этой программы в телеконференции Internet, благодаря чему многие пользователи смогли начать с ней работать.

Популярность алгоритма LZW в значительной степени связана с успехом программы compress. Исходный текст последней версии программы, осуществляющей как сжатие, так и декомпрессию, занимает всего 1200 строк. Ядро кода сжатия занимает не более сотни строк, а код декомпрессии не намного больше. Программисты считают, что это облегчает чтение и понимание алгоритма, а также позволяет адаптировать его для самых разных целей.

Алгоритмы LZ-стиля (включая LZW, LZ77, LZ78 и многие другие варианты) очень популярны везде, где требуется универсальное сжатие. LZW используется в стандарте модема V.42bis, протоколе передачи данных ZModem, форматах GIF, TIFF, ARC и других прикладных программах. Другие алгоритмы LZ используются в дисковых утилитах сжатия типа DoubleSpace и Stacker, графических форматах типа PNG, а также в универсальных утилитах архивирования и сжатия, включая ZIP, GZIP и LHA.

Помимо пользующихся большим вниманием алгоритмов, базирующихся на словаре, существуют и другие подходы. Алгоритм сжатия Хаффмана (Huffman), основанный на статистических колебаниях распределения некоторых значений байтов, лег в основу нескольких очень эффективных методов сжатия, известных, как арифметическое кодирование (arithmetic encoding), энтропийное кодирование (entropy coding) или Q-кодирование (Q-coding). Арифметическое кодирование улучшает сжатие Хаффмана двумя путями. Первое усовершенствование заключается в том, что оно не требует, чтобы выбранные коды были целым числом бит. В то время как сжатие Хаффмана могло выбирать двух- и четырехбитовые коды, программа арифметического кодирования может использовать код длиной 6,23 бит. (Что такое 0,23 бит - чисто философский вопрос, если Вас это заинтересовало, то в отдельном разделе Вы найдете другое объяснение арифметического кодирования.) Второе усовершенствование (которое может

также использоваться в сжатии Хаффмана) заключается в том, что арифметическое кодирование использует более сложную статистику. Она не просто следит за частотой появления байта в файле, а оценивает частоту его появления в определенном контексте. Например, при использовании исходного алгоритма сжатия Хаффмана символ "u", встречающийся не слишком часто, мог бы получать довольно длинный код. Но в сложной программе арифметического кодирования символ "u", следующий за "q", будет закодирован очень компактно, так как высока вероятность того, что "u" следует сразу за "q". Комбинация этих двух усовершенствований приводит очень к эффективному сжатию.

Другие методы сжатия предназначены для данных определенного типа, а потому они плохо подходят для архивирования. Многие усовершенствованные методы, появлявшиеся в последнее время, основывались на синтезе этих трех методов (например, использование кодов Хаффмана для записей словаря) или выполнения сложной предварительной обработки данных, увеличивающей эффективность сжатия одним из этих методов. (Файлы JPEG с выборочно удаленными графическими данными можно затем сжать с помощью метода Хаффмана или арифметического кодирования. Для более эффективного кодирования с помощью методов, базирующихся на словаре, PNG преобразует графические данные, используя для этого простую методику фильтра.)

Возможно, одним из наиболее существенных событий за последние несколько десятилетий в области алгоритмов сжатия стало появление патентов на программное обеспечение. С 1981 United States Patent and Trademark Office (USPTO) начал принимать заявки на патентование алгоритмов программного обеспечения. Многие из представленных патентов были по методам сжатия. Наиболее известные из них - патенты фирмы Unisys на алгоритм сжатия LZW и патенты фирмы IBM на арифметическое кодирование. К сожалению, первоначально работа по обработке заявок в USPTO была поставлена неважно. В результате чего разным людям предоставлялись различные патенты на один и тот же алгоритм (причем иногда с почти идентичной формулировкой). Некоторые из этих патентов оспаривались в судебном порядке, но высокая стоимость судебного разбирательства исков резко снижает количество таких претендентов.

Один положительный результат введения патентования вряд ли приходится оспаривать. Патентование программного обеспечения спровоцировало появление огромного количества работ по разработке новых алгоритмов сжатия (большая часть

которых быстро патентуется их изобретателями). Однако другой эффект был абсолютно отрицательный. Многие из алгоритмов сжатия использовались специфическим образом, например, как часть международных стандартов (V.42bis и JPEG). Кроме того, отдельные компании и пользователи скопировали общедоступный код (так, реализация compress LZW широко копировалась для самых разных целей). Финансовые штрафы за использование этих алгоритмов (в форме авторских отчислений к владельцам патента) отвращали от поддержки этих стандартов авторов условно-бесплатного и бесплатного программного обеспечения или бесплатных библиотек. Некоторые компании публично объявили о том, что они не будут требовать авторских отчислений за использование их запатентованных алгоритмов в бесплатном программном обеспечении. Однако так поступили далеко не все. Пока неясно, как этот конфликт отразится на индустрии бесплатного программного обеспечения и на патентном законодательстве. По крайней мере, одна организация, League for Programming Freedom, борется с патентами программного обеспечения и предпринимает активные шаги по их отмене.

Free Software Foundation (FSF) - организация, работающая над созданием полномасштабной Unix-подобной операционной системы GNU. Наиболее интересный аспект работы FSF состоит в формулировке авторского права на все их программные продукты. Лицензионное соглашение GNU можно охрактеризовать термином "copyleft" (копирование "налево") в противоположность коммерческому термину "copyright" (право копирования). Этот "термин" приводит в недоумение многих людей, перестающих понимать основы лицензионного права. Вопреки удачному названию лицензионного соглашения GNU на самом деле представляет собой именно "copyright". Оно накладывает некоторые ограничения на использование свободно распространяемого программного обеспечения. Лицензия GNU General Public License (GPL) среди прочего гарантирует, что конечные пользователи программного обеспечения FSF будут всегда иметь доступ к исходному тексту этого программного обеспечения. Все, кто используют программные продукты FSF в своих разработках, обязаны обеспечить доступ к исходному тексту своих программ.

Несовершенство сжатия

Алгоритмы сжатия полезны, но и они имеют свои ограничения. Наиболее очевидное ограничение состоит в том, что никакой из методов сжатия (или комбинация методов

сжатия) не совершенен. Некоторые данные могут даже, как это ни парадоксально, в результате применения одной из техник сжатия увеличиваться в размерах. Внимательно смотрите на то, как работает алгоритм сжатия. Заявление о том, что алгоритм может уменьшить в размере файлы с самыми разными типами данных, требует особого внимания. Хорошо написанные программы сжатия помещают в начало выходных данных маркер, указывающий на то, каким методом данные были сжаты. Если данные сжать не удалось, маркер указывает, что данные "не сжаты". При этом размер файла увеличивается на размер маркера.

Иногда программы сжатия утверждают, что они сжимают практически "любой файл на 16 килобайт" или "сжимают каждый файл, по крайней мере, на 30 процентов". Любое такое утверждение просто неверно, хотя некоторые весьма престижные издания стали публиковать объявления о таких программах. В 1992 году фирма WEB Technologies объявила о выходе новой программы сжатия DataFiles/16. При этом фирма сделала ряд интересных заявлений. Прежде всего, она утверждала, что с "помощью неоднократного использвания DataFiles/16 фактически любое количество данных может быть уплотнено до 1024 байт".

Идея неоднократного использования программы сжатия для получения наиболее эффективного сжатия крайне привлекательна. Во многих отношениях эта идея - программистский аналог идеи вечного двигателя; люди никак не могут поверить в то, что это не возможно. Ниже Вы найдете доказательство того, что этого быть, увы, не может. Не один WEB делал подобные заявления, просто его голос оказался наиболее громким. Практически, при вторичном использовании любой хорошей программы сжатия размер будет уменьшаться на пару процентов. После чего повторные попытки будут приводить к обратному эффекту. Так что обещания WEB, к сожалению, остаются лишь обещаниями.

Как показала практика, некоторые из утверждений о суперсжатии любых данных были чистейшей воды обманом: часть якобы сжимаемых данных просто копировалась в некий невидимый файл, то же, что оставалось, конечно, было меньшего размера. В то время как такая схема выглядит внушительной в распечатке каталога, ее едва ли можно квалифицировать как "сжатие". Как правило, такие профанации без шума исчезали, и никто так и не исследовал этих "совершенных" методов сжатия.

Некоторые из утверждений о совершенном сжатии на поверку оказывались простым мошенничеством. Одна из таких программ, сделав запрос на архивирование, удаляла файл с жесткого диска, увеличивая при этом архив лишь на сто байт, а деархиватор восстанавливал его, если только вы не слишком много работали с жестким диском между этими двумя событиями. Программа фактически хранила только имя файла и его расположение на жестком диске, а по завершении своей "работы" она удаляла этот файл с жесткого диска. Это приводило к правильному восстановлению файла каталога только в том случае, когда эта часть жесткого диска не использовалась многократно при работе с другими файлами. В противном же случае ваши данные просто исчезали. Чтобы выявить обман такого рода, необходимо провести следующий эксперимент: скопируйте несколько файлов на только что отформатированную дискету, используйте программу архивации файлов, а затем скопируйте архив на другую только что отформатированную дискету и попробуйте разархивировать полученный таким образом файл.

Не трудно догадаться, почему любая методика сжатия должна увеличивать размеры некоторых файлов. Не забывайте о том, что это в действительности - техника "кодирования", которая берет некоторую информацию и хранит ее по-новому. Предубеждение против использования слова "сжатие" базируется лишь на том, что любой метод кодирования, уменьшающий объем одной информации, должен увеличивать объем другой.

Реальным доказательством работы метода является не кодирование (сжатие), а декодирование (распаковка). Метод сжатия, не позволяющий восстанавливать первоначальные данные, вряд ли полезен.

Проделаем небольшой мысленный эксперимент. Представьте, что в Вашем распоряжении оказалась программа сжатия, которая действительно уменьшает размер каждого файла. Также представьте, что Ваш компьютер имеет действительно большой жесткий диск. на котором хранятся всевозможные различные файлы длиной 10000 байт. Теперь предположим, что вы с помощью Вашей гипотетической программы сжатия сжали каждый из этих файлов. После завершения этой работы каждый из файлов стал короче 10 000 байт.

Это может показаться Вам несущественным, но все же давайте точно определим, сколько на жестком диске файлов. Так как в каждом байте восемь бит, то у Вас должно быть 2 в степени 80 000 файлов, причем размер каждого из них точно 10 000 байт. В

результате мысленного эксперимента у нас по-прежнему останется 2 в степени 80 000 файлов, каждый из которых будет короче чем 10 000 байт. Однако не совсем понятно, каким образом могут получиться два совершенно оди-наковых "сжатых" файла! А ведь именно это мы и получаем, поскольку не так уж и много существует файлов короче 10000 байтов. Предположив, что полученные файлы отличаются друг от друга по размеру всего на один байт, то есть 9 999 байтов, затем 9 998 байтов и так далее, вы получите меньше чем 2 в степени 80 000 возможных вариантов.

Таким образом, вы приходите к важному выводу, что по крайней мере два сжатых файла должны быть идентичны. Если не вся логическая цепочка, приводящая к этому выводу, для Вас очевидна, представьте, что у Вас есть пять перевернутых карт. Так как существуют только четыре варианта масти, вы знаете, что две из них имеют одинаковую масть (возможно, что все пять карт имеют одинаковую масть, но Вы не можете об этом знать наверняка). В данном случае применялся тот же принцип: эксперимент предполагал наличие 2 в степени 80 000 файлов, однако после сжатия возможное количество неодинаковых файлов стало меньше этого значения. Так как их число не должно было измениться, то по меньшей мере два файла должны быть одинаковы.

Что же все это значит? Вы начали с предположения о том, что в вашем распоряжении есть совершенный метод сжатия, который должен сжимать каждый из файлов. В результате эксперимента стало очевидно, что по крайней мере два файла, различавшихся до сжатия, стали совершенно одинаковыми после сжатия. Однако методов, позволяющих из двух идентичных файлов в результате декомпрессии получить два разных оригинала, не существует!

Таким образом, этот мысленный эксперимент доказывает, что программы, которая сжимала бы каждый файл и могла бы затем при необходимости все корректно распаковать, не существует. Конечно, программа сжатия, не позволяющая вам восстанавливать исходные данные, практически бесполезна. (Чтобы уменьшить размер газеты, вы можете ее сжечь, однако после этого узнать, что в ней было написано, будет довольно сложно.)

Сжатие Чарльза Уэзерелла

Все знают, что большинству людей свойственно излишнее многословие. Гораздо менее широко известно, что даже самые лаконичные высказывания можно было бы

значительно сократить. Вообще, естественные языки отличаются чрезвычайной избыточностью. «Даж есл нсклко бкв вбрсть, эт прдлжн ещ мжн прчть.» Языки, используемые для вычислений, обладают той же особенностью. Для экономии памяти компьютера, объем которой ограничен, имеет смысл ликвидировать избыточность текста.

Существует несколько способов уплотнения текста. Самый очевидный из них - поиск различных по длине цепочек из одной повторяющейся литеры. Такая группа может быть заменена тройкой литер mcn, где m обозначает признак повторения, специальную литеру, не используемую нигде в тексте для других целей, c - сама повторяющаяся литера и n - длина цепочки. Один такой триграф экономит n - 3 литер, причем значение n не может превышать максимального числа, представимого в поле одной литеры. Описанный способ обработки весьма неплохо оправдывает себя для текстов, содержащих длинные цепочки повторяющихся литер, например длинные цепочки пробелов, характерных для большинства программ. К сожалению, этот прием не столь хорош для других текстов, поскольку большинство данных не отличается такой же строгой формой записи, как программы.

Второй способ основан на том, что в различных системах кодировки литер, применяемых на ЭВМ, большинство литер практически не используется (из 256 литер обычного 8-разрядного кода, как правило, употребляется лишь около 100). Сначала в тексте отыскиваются наиболее распространенные диграфы, и каждому из них ставится в соответствие одна из не используемых в тексте одиночных литер. Уплотнение текста производится при просмотре его слева направо путем последовательной замены выявленных диграфов их однолитерными эквивалентами. При этом может быть достигнута значительная экономия, поскольку, например, 150 наиболее часто встречающихся диграфов уже составляют большую долю текста на естественном языке. И если не ставить целью слишком высокую степень уплотнения текста, можно написать довольно эффективные программы кодирования и деко-дирования, работающие с машинным представлением литер.

Однако существуют все же определенные трудности. Кто сказал, что наиболее часто встречающиеся диграфы в английском тексте должны быть теми же, что и во французском, или в наборе файлов, содержащих почтовые адреса, или в тексте на Алголе? А если даже это и так, то как насчет триграфов, квадриграфов или более

длинных групп? Ведь более длинные группы, даже если они и реже встречаются, дают большую экономию, а бывает, что определенный фрагмент появляется в большом куске текста намного чаще, чем можно было бы ожидать. И, возвращаясь назад, как подсчитать частоты появления диграфов?

Ответ на все эти вопросы содержится в третьем подходе к решению исходной задачи. Вместо того чтобы употреблять некоторый, заранее заданный набор кодировок, можно на ходу генерировать кодовый словарь, используя непосредственно текст, подлежащий сжатию, или выборку из него. Поскольку при этом каждый элемент текста будет участвовать в создании своего собственного словаря, исчезнут трудности, вызванные неудачными аббревиатурами. Теперь нам надо найти способ построения такого словаря.

Опишем наш план действий в общих чертах. Начинаем с пустого словаря. Текст просматриваем слева направо. Ищем в словаре гнездо возможно большей длины, совпадающее с головной частью текста, и увеличиваем счетчик частоты соответствующего гнезда словаря. Если совпадений цепочек нет, образуем новое гнездо словаря и помещаем туда первую букву текста. Вычеркиваем обработанную цепочку из начала текста и начинаем просмотр заново. При обстоятельствах, поясняемых ниже, иногда два гнезда словаря соединяются в одно, образуя цепочку большей длины - процесс укрупнения гнезд. Когда словарь переполняется, производим его чистку, удаляя наиболее редко встречающиеся гнезда, и продолжаем просмотр. После того как частоты встречаемости гнезд словаря стабилизируются, вводим таблицу кодировок и, взяв исходный текст, полностью его кодируем.

В предложенной схеме есть два невыясненных момента: каким образом происходит укрупнение гнезд словаря и как осуществляется его чистка? Укрупнение двух гнезд словаря производится в случае, когда одно из них следует в тексте непосредственно за другим и частоты обоих гнезд превышают некоторое пороговое значение. При этом, чтобы новое гнездо словаря не подвергалось ближайшей чистке, ему может быть приписана начальная частота несколько выше обычной. Таким образом, если в словаре уже имеются, например, цепочки КОН и ТАКТ, то при условии, что содержимое их счетчиков достаточно велико, может образоваться новое гнездо словаря, содержащее цепочку КОНТАКТ. Что же касается чистки словаря, то существует простой способ - удалять все те гнезда, значения счетчиков которых меньше среднего. Можно действовать

и иначе - выбрасывать все гнезда, частота которых ниже медианы частот. Годятся и другие, подобные этому способы.

Алгоритм построения словаря

В приводимом алгоритме предполагается, что построение словаря производится с помощью некоторой выборки из текста, подлежащего сжатию. Для алгоритма существенны все литеры текста, и если табуляция, концы строк и другие аналогичные элементы имеют значение, то в тексте должны присутствовать соответствующие управляющие литеры. Предполагается, что в начале работы словарь пуст. В начальный момент переменная last match содержит пустую цепочку, а переменная last count имеет значение, равное нулю.

1. Ищем в головной части входного текста возможно более длинную цепочку match, совпадающую с каким-нибудь гнездом словаря. Если переменная match пустая, засылаем в нее первую литеру входного текста, помещаем в свободное гнездо словаря и устанавливаем начальное значение счетчика этого нового гнезда равным единице. Если цепочка match не пустая, увеличиваем на единицу счетчик соответствующего гнезда словаря. Содержимое счетчика этого гнезда записываем в count.

2. Если либо count, либо last count меньше значения порога укрупнения гнезд, то переходим к шагу 4. Порог укрупнения определяется как отношение максимально допустимого объема словаря к числу оставшихся в данный момент свободных гнезд.

3. Образуем новое гнездо словаря путем объединения цепочек last match и match. Поскольку данное гнездо словаря возникло впервые, засылаем в его счетчик единицу. Можно применить и другие стратегии.

4. Если в словаре остались свободными менее двух гнезд, производим чистку, удаляя все гнезда с частотами меньше медианы частот. При этом, если окажется, что исключилось гнездо, содержащее match, устанавливаем count равным нулю.

5. Вычеркиваем match из начала входного текста. Если текст исчерпан, то алгоритм работу заканчивает - выход. В противном случае помещаем last match в match, пересылаем last count в count и возвращаемся к шагу 1.

Кодирование и декодирование

Как только построение словаря завершилось, необходимо составить таблицы для кодирования и декодирования. Образуем все возможные диграфы, начинающиеся с литеры, которая нигде в тексте не используется. Исключим из словаря все гнезда, состоящие из одной или двух литер (их уплотнение экономии дать не может). Упорядочим оставшиеся цепочки по частоте встречаемости. Поставим в соответствие гнездам словаря полученные выше кодирующие диграфы, начиная с гнезд, имеющих наибольшую частоту. Формирование таблицы кодировок завершается по исчерпании гнезд словаря или набора диграфов.

Процесс кодирования текста подобен процедуре построения словаря. На каждом этапе головная часть входного текста проверяется на совпадение в возможно большем числе позиции с гнездами словаря. Совпавшая цепочка заменяется в тексте соответствующим кодирующим диграфом, и начало просмотра входного текста сдвигается на длину выделенной цепочки. Если же в словаре не найдено нужного гнезда, в выходной текст просто переносится первая литера из головной части входного текста и начало просмотра перемещается вправо на одну позицию. Декодирование осуществляется путем простой замены кодирующих диграфов их эквивалентами из словаря.

Тема. Напишите программу, реализующую описанные выше алгоритмы построения словаря, кодирования и декодирования. Проверьте программу на достаточно больших фрагментах текста на естественном языке и языке программирования. Коэффициент сжатия данного куска текста определяется как частное от деления суммы длин сжатого текста и словаря на длину исходного текста. Проведите небольшое исследование зависимости коэффициента сжатия от какого-нибудь из следующих параметров: языка уплотняемого текста; объема используемой для упражнения выборки из текста; длины словаря при его построении; имеющегося количества кодирующих диграфов или применимости словаря, полученного на основании одного текста, для другого текста на том же языке.

Указания исполнителю. Данная задача интересна тем, что для ее эффективного решения требуется употребить некоторые весьма развитые алгоритмы и структуры данных. Однако пусть не столь эффективную, но правильно работающую программу можно написать, используя простые алгоритмы и структуры, которые можно. когда программа заработает, постепенно заменять более изящными конструкциями. Одним из

примеров служит вычисление медианы для чистки словаря. В качестве первого варианта можно просто выбрасывать гнезда словаря с частотами, меньшими средней. При этом среднюю частоту легко вычислить за один полный просмотр всех частот словаря. А после того как такая программа в целом заработает, можно уже для нахождения порога исключения строк подключить более сложную программу расчета медианы.

Другим примером является выбор структуры словаря на этапах его создания и кодирования. Если гнезда словаря расположить в случайном порядке, то при проверках на совпадение необходимо проходить весь словарь. Однако при такой структуре появляющиеся новые гнезда добавляются просто в конец словаря. Небольшое усложнение могло бы заключаться в группировке гнезд словаря по их длинам. Тогда поиск мог бы осуществляться в направлении от самых длинных групп к коротким и прекращаться при первом же удачном сравнении. Если каждую группу еще и лексикографически упорядочить, то можно было бы воспользоваться вместо линейного поиска внутри группы двоичным поиском, экономя таким образом время. Но зато добавление в словарь новых гнезд становится в этом случае более сложным, так как для любого нового гнезда потребуется место, скорее всего, где-то в середине группы. Не исключено, что самой выгодной структурой для организации поиска окажется какая-либо разновидность дерева. Разыскиваемую цепочку словаря могла бы тогда составить последовательность букв по пути от корня дерева к его листьям, или, иначе говоря, в узлах некоего подобия двоичного дерева поиска могли бы располагаться соответствующие строки словаря. В то же время при составлении словаря деревья потребуют намного большей обработки, нежели описанные выше более простые структуры.

Инструментовка. Вследствие разнообразия структур данных, используемых в готовой программе, исходный язык должен обладать хорошими средствами описания данных. В этом плане можно рекомендовать Паскаль, Алгол-68 и PL/I. Можно было бы предложить сначала написать программу на Сноболе, опираясь на заложенные в этом языке средства сопоставления с образцом, а затем переписать готовую программу на каком-нибудь более эффективном при массовых расчетах языке. При использовании этого пути необходимо быть внимательными и избегать употребления таких средств Снобола, которые трудно воспроизвести на другом языке.

Длительность исполнения. Одному исполнителю на 3 недели.

Развитие темы. В описанной модели имеются три области свободы: критерий укрупнения гнезд, критерий исключения низкочастотных гнезд словаря и система их кодирования. Рассматривая их по порядку, начнем с критерия укрупнения гнезд. Для того чтобы могло произойти укрупнение гнезд, в нашем алгоритме требуется, чтобы частоты встречаемости каждого из двух последовательных гнезд превысили один и тот же крайний предел.

Можно, однако, для каждого гнезда иметь свой порог. В другом варианте может быть у одного гнезда постоянный порог, а у другого - порог, являющийся функцией средней частоты гнезд. Аналогично, может варьироваться начальная частота укрупненного гнезда, причем при любом способе начальная частота задается большой, исходя из условия повышения шансов на сохранение данного гнезда при чистке.

Точно так же может быть видоизменен образ действий при исключении гнезд словаря во время его чистки. Можно выбрасывать неизменную часть низкочастотных гнезд (используя медиану, устанавливающую эту часть равной половине). Можно исключать все гнезда с частотами, меньшими некоторой, кратной средней частоте. Или же можно вычеркивать все гнезда с частотами, меньшими заданной, и эту процедуру осуществлять до тех пор, пока словарь не будет достаточно вычищен. Сочетание различных способов укрупнения и чистки гнезд характеризуется особым показателем исключаемости. В некоторых вариантах оставляются цепочки, которые часто встречаются в одной части текста и реже в других; в иных случаях предпочтение отдается цепочкам, равномерно разбросанным по тексту. Какому показателю исключаемости отдать предпочтение, зависит от используемых особенностей как словаря, так и текста.

В алгоритме кодирования употребляются диграфы, начинающиеся с не используемых во входном тексте литер. Однако, если набор диграфов кончился, а словарь еще не доделан, можно использовать триграфы и т. д. Коль скоро частоты гнезд словаря известны, их можно употребить для организации взвешенного кодирования переменной длины. Этот способ будет дороже при декодировании (почему не при кодировании?), зато обеспечит даже более высокую степень сжатия текста.

МЕТОД ХАФФМАНА И РОДСТВЕННЫЕ МЕТОДЫ

Метод Хаффмана (Huffman method)

Сжатие Хаффмана - статистический метод сжатия, который уменьшает среднюю длину кодового слова для символов алфавита. Код Хаффмана является примером кода, оптимального в случае, когда все вероятности появления символов в сообщении - целые отрицательные степени двойки. Код Хаффмана может быть построен по следующему алгоритму:

1. Выписываем в ряд все символы алфавита в порядке возрастания или убывания вероятности их появления в тексте;

2. Последовательно объединяем два символа с наименьшими вероятностями появления в новый составной символ, вероятность появления которого полагается равной сумме вероятностей составляющих его символов; в конце концов мы построим дерево, каждый узел которого имеет суммарную вероятность всех узлов, находящихся ниже него;

3. Прослеживаем путь к каждому листу дерева помечая направление к каждому узлу (например, направо - 1, налево - 0).

Для заданного распределения частот символов может существовать несколько возможных кодов Хаффмана. Возможно определить 'каноническое' дерево Хаффмана, выбрав одно из возможных деревьев. Такое кононическое дерево может быть очень компактно, передавая только длину в битах для каждого кодового слова. Такой метод используется в большинстве архиваторов (pkzip, lha, zoo, arj, ...).

Метод Шеннона-Фано (Shannon-Fano method)

Родственным методом для кодирования Хаффмана является кодирование Шеннона-Фано, которое осуществляется следующим образом:

1. Делим множество символов на два подмножества так, чтобы сумма вероятностей появления символов одного подмножества была примерно равна сумме вероятностей появления символов другого. Для левого подмножества каждому символу приписываем "0", для правого - "1".

2. Повторяем шаг (1) до тех пор, пока все подмножества не будут состоять из одного элемента.

Алгоритм создания кода Хаффмана называется снизу-вверх, а Шеннона-Фано - сверху-вниз. Кодирование по Хаффману всегда дает оптимальные коды, по Шеннону-Фано иногда используется немного больше бит.

Алгоритмы Хаффмана и Шеннона-Фано являются одними из классических, поэтому они часто используются в графических форматах. Они берут только частоту появления одинаковых байт в изображении и сопоставляют символам входного потока, которые встречаются большее число раз, цепочку бит меньшей длины. И напротив - встречающимся редко - цепочку большей длины. Для сбора статистики требуют двух проходов по изображению. Коэффициенты сжатия алгоритмов: 1/8, 2/3, 1. Требуют записи в файл таблицы соответствия кодируемых символов и кодирующих цепочек.

На практике используются их разновидности. Так, в некоторых случаях резонно либо использовать постоянную таблицу, либо строить ее адаптивно, т.е. в процессе архивации/разархивации. Эти приемы избавляют от двух проходов по изображению и необходимости хранения таблицы вместе с файлом. Кодирование с фиксированной таблицей применяется в качестве последнего этапа архивации в JPEG.

Близкая модификация алгоритма используется при сжатии черно-белых изображений. Последовательности подряд идущих черных и белых точек заменяются числом, равным их количеству с признаком цвета. А этот ряд уже, в свою очередь, сжимается по Хаффману с фиксированной таблицей. Для того, чтобы понять использование этих двух методов для сжатия изображений, рекомендую обратиться к формату TIFF.

Арифметический метод (Arithmetic method)

Может показаться, что кодирование Хаффмана или Шеннона-Фано лучшее средство для сжатия. Однако это не так. Как было замечено выше, эти методы оптимальны только в том случае, когда все символы в сообщении имеют вероятности появления равные целым отрицательным степеням двойки, что в общем случае не так.

Метод арифметического кодирования не имеет этого ограничения: он достигает одинакового эффекта, т.к. рассматривает сообщение как единое целое (что для кодирования по Хаффману потребовало бы нумерации каждого из всех возможных сообщений), и таким образом достигает теоретической энтропийной границы эффективности сжатия для любого источника.

Работа арифметического кодера состоит в представлении числа интервалом вещественных чисел от 0 до 1. По мере увеличения длины сообщения, интервал, необходимый для его представления, становится все меньше и меньше, а число бит, необходимых для задания этого интервала, увеличивается. Каждый символ сообщения по порядку сокращает этот интервал прпорционально вероятности появления этого символа. Наиболее вероятный символ меньше всех сокращает интервал, и таким образом добавляет меньше бит к коду сообщения.

В качестве примера арифметического кодирования рассмотрим такой пример: алфавит состоит из двух символов X и Y с вероятностями 0.66 и 0.33. Кодируя такое сообщение, посмотрим на первый символ: если это символ X, выбираем интервал (0; 2/3), если же это символ Y, то - (2/3; 1). Продолжая таким образом для двух символов, мы получим : XX - (0; 4/9), XY - (4/9; 6/9), YX - (6/9; 8/9) и для YY - (8/9; 1).

В этом примере арифметический код не полностью эффективен, по причине сжатия короткого сообщения - на длинных сообщениях эффективность кодирования реально приближается к 100%.

Теперь когда у нас есть эффективный метод кодирования, что мы можем сделать с ним? То что нам надо - метод построения модели данных, которую мы можем затем использовать вместе с кодером. Простейшей моделью, например, является фиксированная таблица вероятностей стандартных букв английского текста, которую мы можем использовать для получения вероятностей букв. Улучшением этого метода является использование адаптивной модели, другими словами, модели, которая для сжатия последующих данных приспосабливает себя по уже сжатым данным. Мы можем преобразовать фиксированную модель в адаптивную изменяя вероятности символов после кодирования каждого нового символа. Однако, мы можем сделать больше, чем это.

Использование вероятностей символов по отдельности - не совсем хорошая оценка для энтропии данных: мы можем также использовать вероятности сочетаний символов. Лучшие из известных на сей момент компрессоры, использующие этот подход: DMC (Dynamic Markov Coding) начинает работу с Марковской модели 0-порядка и постепенно расширяет начальную модель в процессе сжатия; PPM (Prediction by Partial Matching), ищет появления сжимаемого текста в контексте n-го порядка. Оба метода, таким образом, получают более лучшую модель сжимаемых данных, и в сочетании с арифметическим кодированием, приводят к лучшей степени сжатия.

Итак, если кодеры, основанные на арифметическом кодировании, так хороши, почему они не используются широко? Не считая того, что они относительно новы и не успели войти в повсеместное использование, есть еще один существенный недостаток: они требуют гораздо больше вычислительных ресурсов, как ресурсов процессора, так и памяти. Построение сложных моделей для сжатия может не закончится из-за нехватки памяти (особенно в случае DMC, когда модель может неограниченно расти); и само арифметическое кодирование требует решения числовых задач большого объема.

СУПЕРАДАПТИВНОЕ СЖАТИЕ

При описании методов сжатия я заметил, что они ориентированны на файлы, с различным распределением информации. Например, метод Хаффмана дает наилучшие результаты при значительном отличии частот появления символов в файле, а метод Лемпела-Зива основан на замене длинных последовательностей символов более короткими кодами.

Но каждую технику сжатия можно проклассифицировать еще глубже. Для сжатия данных используются как статистические, так и эвристические алгоритмы. Статистические алгоритмы (Хаффмана, Шеннона-Фано, арифметические) требуют для своей работы знания частот появления символов в файле. Однако при однопроходной обработке файла, невозможно одновременно определить частоты символов и получить сжатый файл. Поэтому все статистические алгоритмы можно разделить на три класса:

неадаптивные - используют фиксированные, заранее заданные вероятности символов. Таблица вероятностей символов не передается вместе с файлом, т.к. она заранее известна. Недостаток: узкий класс файлов, для которых достигается приемлемый уровень сжатия.

полуадаптивные - для каждого файла строят таблицу частот символов и с ее помощью сжимают файл. Вместе со сжатым файлом передается таблица символов. Такие алгоритмы достаточно хорошо сжимают большинство файлов, но требуется дополнительная передача таблицы частот символов.

адаптивные - начинают работать с фиксированной начальной таблицей частот символов (обычно все символы изначально одинаково вероятны) и в процессе работы это таблица изменяется в зависимости от встречаемых символов файла. Преимущества: однопроходность алгоритма; также как и неадаптивные алгоритмы, не требуется

передача таблицы частот символов, но достаточно эффективно сжимается широкий класс файлов.

Эвристические алгоритмы сжатия (типа LZ77, LZ78), как правило, ищут повторяющиеся строки в файле, либо строят словарь как уже встречавшихся фраз, так и фраз, которые наиболее вероятно могут появиться в тексте. Современные версии этих алгоритмов сами по себе уже являются адаптивными (они составляют словарь из последовательностей, встретившихся в файле), но одни из первых программ эвристического сжатия основывались на заранее заданном словаре последовательностей и чаще всего были настроены на сжатие определенных типов данных (текстовых файлов, баз данных, графических данных).

Обычно такие алгоритмы обладают целым рядом специфических параметров (размер буфера, максимальная длина фразы, размер рассматриваемого контекста и т.п.), подбор которых зависит от опыта автора алгоритма, и эти параметры подбираются так, чтобы добиться оптимального сочетания времени работы, коэффициента сжатия и широты класса хорошо сжимаемых файлов.

При подборе этих параметров можно заметить, что для различных файлов (текстовые, двоичные, базы данных) оптимальны различные сочетания параметров, не говоря уже о том, что разные алгоритмы оптимальны для разных классов исходных файлов.

Идея суперадаптивного сжатия заключается в адаптивности параметров сжимающего алгоритма, т.е. параметры алгоритма, или же сами алгоритмы сжатия, могут меняться в зависимости от текущего распределения частот символов в обрабатываемом файле. Алгоритм при этом остается однопроходным.

Вся тонкость заключается в решении о принадлежности данного файла к тому или иному классу. Можно заметить, что для файлов того или иного класса свойственно определенное распределение частот символов. Конечно, для каждого файла распределение частот уникально, но у файлов одного класса эти распределения не сильно различаются. Таким образом, проблема сводится к определению вида распределения символов, к которому ближе всего в данный момент распределение символов обрабатываемого файла.

Мерой близости одного распределения частот символов к другому может служить количество информации. В отличие от Шенноновской энтропии, существует несколько

различных определений количества информации. Это связано с условиями, которые накладываются на нее. Этим условиям удовлетворяет несколько различных функций. Не вдаваясь в математические подробности, скажем самое главное: эта функция должна обращаться в ноль при равенстве распределений и должна быть положительной в других случаях.

Примерная схема суперадаптивного сжатия:

Нам известны несколько различных алгоритмов сжатия и распределения частот для которых они оптимальны. Стартуем с фиксированного распределения частот и соответствующего ему алгоритма.

Используя текущий алгоритм кодируем N очередных символов текста, изменяем таблицу частот встреченных символов, подсчитывает количество информации для всех известных распределений, определяем распределение к которому ближе всего текущее распределение символов (для него значение количества информации наименьшее), выбираем соответствующий ему алгоритм. И опять повторяем тоже самое для всего файла.

Еще одним способом суперадаптации программ сжатия является объединение нескольких алгоритмов, сжимающих различные типы данных. Наилучший пример таких алгоритмов - алгоритм, использованный в известном архиваторе ZIP Фила Каца. Объединение алгоритма LZ77 и Хаффмана дает великолепный результат.

Еще дальше пошли разработчики современных графических стандартов: например, в стандарте JPEG наряду с алгоритмом Хаффмана используется логическая сортировка графических данных (еще один способ сжатия).

МЕТОД ЛЕМПЕЛА-ЗИВА И РОДСТВЕННЫЕ МЕТОДЫ

Семейство LZ-компрессоров. Замещающие компрессоры.

Основная идея, используемая в замещающих компрессорах, состоит в замене появления определенной фразы или группы байт во фрагменте данных ссылкой на предыдущее появление этой фразы. Существуют два основных класса методов, названных в честь Якоба Зива (Jakob Ziv) и Абрахама Лемпеля (Abraham Lempel), впервые предложивших их в 1977 и 1978.

Алгоритм LZ78

Алгоритм LZ78 вносит все встреченные им последовательности в словарь. Всякий раз, когда среди данных, которые надо сжать, встречается последовательность, программа обращается к словарю:

- если последовательность находится в словаре, то в выходной файл заносится код для этой записи;

- если последовательность представляет собой расширенный вариант последовательности из словаря, то она добавляется в таблицу.

Например, если программа сжатия уже имеет последовательность ABC в словаре и увидит последовательность ABCA, то она вначале занесет в выходной файл код из словаря для ABC, затем для символа A, после чего последовательность ABCA будет добавлена в словарь. Если же она позже встретит ABCAB, то выведет код для ABCA, затем код для символа B и добавит ABCAB в словарь. Каждый раз, когда программа встречает последовательность из словаря, она выдает код и добавляет новую запись, которая на один байт длиннее. Таким образом, каждый раз при повторении последовательности байтов, словарь должен будет расти, чтобы включить продолжение этой последовательности. Обратите внимание на то, что программа должна встретиться с последовательностью ABCABC по крайней мере 6 раз (по числу букв), прежде чем в словаре будет создана запись, содержащая всю последовательность.

Фактически, предшествующий абзац немного искажает реальную картину. Программа сжатия обрабатывает в каждый момент времени лишь один, а не несколько байтов, хотя конечный результат остается по-прежнему таким же. Первоначально словарь содержит всевозможные однобайтовые значения, пронумерованные от 0 до 255, и одну дополнительную запись с номером 256 (о ней я расскажу немного позже). Давайте посмотрим, как программа сжатия LZ читает каждый байт abcabc.

a Уже есть в словаре, и программа сжатия знает об этом. Поэтому она запоминает этот символ и переходит к следующему байту.

b Последовательности ab в словаре нет, поэтому она туда добавляется с присваиваемым ей кодом 257. Программа выводит в выходной файл a и начинает накапливать последовательность, начинающуюся с b.

c Последовательности bc в словаре нет, поэтому она туда добавляется с кодом 258. Программа выводит b и переходит к поиску

последовательности, начинающейся с c.

a Опять-таки последовательности ca в словаре нет, поэтому она туда добавляется с кодом 259. Программа выводит c и переходит к поиску последовательности, начинающейся с a.

b Теперь последовательность ab в словаре уже есть, программа помнит об этом (фактически она помнит, что код этой последовательности 257), заносит в выходной файл этот код и продолжает двигаться дальше.

c Теперь программа имеет код 257 (код для ab) и c. Так как последовательности abc в словаре нет, она туда добавляется с кодом 260, выводится код 257, и программа ищет дальше последовательность, начинающуюся с c.

Таблица 1.

Пример сжатия

Предыдущий код	**Текущий байт**	**Текущая последовательность**	**Добавляется в словарь**	**Выходной файл**
Нет	a	a		
A	b	ab	257	a
B	c	bc	258	b
C	a	ca	259	c
A	b	ab(257)		
257	c	abc	260	257
C	a	ca(259)		
259	b	cab	261	259
b	c	bc(258)		
258	a	bca	262	258
a	b	ab(257)		
257	c	abc(260)		
260	a	abca	263	260
A	b	ab(257)		
257	c	abc(260)		
260	a	abca(263)		
263	b	abcab	264	263

В таблице 1 Вы найдете более длинный пример работы алгоритма, представленный в более компактной форме. Обратите внимание, что во время каждого шага проверяется, есть ли текущая последовательность в таблице (код указывается в круглых скобках), и если ее там нет, она туда добавляется. Также обратите внимание на то, что последовательности в таблице становятся все длиннее и длиннее, а количество данных в столбце "выходные данные" становится все меньше и меньше. Следует заметить, что это происходит только в случаях подобных данному, когда последовательности часто повторяются.

Если вы внимательно проследите за выполнением каждого шага, то легко поймете, как работает алгоритм сжатия LZ. Кроме словаря, программа фактически использует очень мало данных. Ей требуется только хранить код сравниваемой последовательности и текущего байта.

Но программа сжатия мало полезна без соответствующего распаковщика, так что давайте теперь посмотрим, как работает LZ-распаковщик. Фактически, вы уже располагаете достаточным количеством информации для того, чтобы догадаться, как это происходит. Распаковщик работает по тому же принципу, что и программа сжатия. Всякий раз, когда программа сжатия находит длинную последовательность, которую необходимо добавить в словарь, она, прежде чем это сделать, выводит предыдущий код. Распаковщик может только следовать дальше, строя такой же словарь, что и программа сжатия. Всякий раз, когда распаковщик читает код, он декодирует его по словарю и затем подражает действиям программы сжатия, чтобы модифицировать словарь. Таблица 2 показывает, как распаковщик декодирует выходные данные программы сжатия, полученные нами в предыдущем примере. Обратите внимание на то, что он формирует тот же самый словарь, что и программа сжатия.

Таблица 2.

Пример распаковки

Входные данные	**Выходные данные**	**Добавляется в словарь**
A	a	
B	b	ab(257)
C	c	bc(258)
257	ab	ca(259)
259	ca	abc(260)

258	bc	cab(261)
260	abc	bca(262)
263	abca	abca(263)

Как видно из таблицы 2, распаковщик никогда не встречает код, который не был бы уже записан в словарь. Исключение составляют данные, имеющие вид байт - последовательность - байт - последовательность - байт типа a bc a bc a. В этом и только в этом случае код для байт - последовательность - байт (в нашем примере это 263) будет встречен программой декодирования до того, как он попадет в словарь, поэтому распаковщик может использовать предыдущий код (в нашем примере это 260) и на основании его распознать текущий код.

Один из моментов, требующих некоторых дополнительных разъяснений - зарезервированный код 256. Как видно из предыдущих примеров, пока данные сжимаются, объем словаря устойчиво растет. При использовании этого алгоритма для сжатия файлов, имеющих очень большие размеры, вы должны каким-то образом ограничить объем памяти, выделяемой для хранения словаря. Программа сжатия устанавливает ограничение на размер словаря (обычно это значение находится в пределах 4 096 - 65 536 записей), и после того, как объем достигает этого значения, она очищает словарь. Таким образом, может быть ограничен объем используемой памяти. Чтобы этим не сбить с толку распаковщика, программа сжатия всякий раз, очищая словарь, вставляет код 256, который называется кодом сброса (reset code). Когда распаковщик видит код 256, он сбрасывает словарь.

Когда алгоритм LZ начинает работу, в его словаре всего 257 записей. Таким образом, для представления любого кода словаря требуется только девять бит. После того как количество записей в словаре достигнет 512, потребуется уже десять бит. Алгоритм оптимизирован таким образом, чтобы использовать ровно столько бит, сколько требуется для предоставления всех записей словаря. После того как в словаре появляется 511-я запись, на каждый код будет отводиться 10 бит; с появлением 1023-й записи на каждый код будет использоваться 11 бит и так далее, количество бит будет увеличиваться лишь по мере необходимости. Распаковщик при построении такого же словаря будет также изменять в нужный момент размеры кода.

Алгоритм LZ77

Методы LZ77 отслеживают последние обработанные n байт данных, и когда встречается фраза, которая уже была, вместо нее выдается пара значений, соответствующая позиции фразы в буфере предыстории и длине фразы. В сущности, компрессор передвигает окно фиксированного размера по потоку данных (в основном называемое скользящим окном), и позиция в паре (позиция, длина) обозначает позицию фразы внутри этого окна. Наиболее используемые алгоритмы происходят от метода LZSS, описанного Джеймсом Сторером (James Storer) и Томасом Шиманski (Thomas Szymanski) в 1982. В этом компрессоре используется окно размером N байт и буфер предпросмотра, содержимое которого ищется в окне.

Расжатие просто и быстро: Если встречается пара (позиция, длина), на выход копируется (длина) байт из окна начиная с позиции (позиция).

Методы, использующие "скользящее окно", могут быть упрощены нумеруя входной текст по модулю N, создавая таким образом кольцевой буфер. "Скользящее окно" автоматически создает LRU эффект, который должен быть явно реализован в LZ78 методах. Варианты этого метода применяют дополнительное сжатие выхода LZSS компрессора, включая простой код переменной длинны (LZB), динамическое кодирование Хаффмана (LZH), и кодирование Шеннона-Фано (ZIP 1.x)), все они дают некоторый выигрыш по сравнению с основной схемой, особенно когда данные несколько случайны и компрессор LZSS дает небольшой эффект.

Позднее был изобретен алгоритм, комбинирующий идеи LZ77 и LZ78, и называемый LZFG. LZFG использует стандартное "скользящее окно", но хранит данные в модифицированной древовидной структуре данных и выдает в качестве кода позицию текста в дереве. Т.к. LZFG только вставляет целые фразы в словарь, он должен работать быстрее любого другого компрессора LZ77.

Все популярные архиваторы (arj, lha, zip, zoo) являются вариациями на тему LZ77.

Алгоритм LZW, модернизированный Терри Велчем

Как уже упоминалось, Терри Велч работал над созданием программы compress вместе с группой программистов Unix. При реализации compress он несколько усовершенствовал исходный алгоритм LZ78. Наиболее интересное усовершенствование - адаптивный сброс (adaptive reset). Вместо того, чтобы очищать словарь по достижении определенного объема памяти для его хранения (вариант, рассмотренный до этого),

программа compress продолжает использовать словарь до тех пор, пока остается высоким уровень сжатия. Этот подход базируется на двух наблюдениях, касающихся процесса сжатия LZW. Одно из них заключается в том, что на первых порах величина сжатия сильно зависит от размера словаря. Больший словарь содержит более длинные последовательности, которые в результате эффективно сжимаются в одиночный код. Адаптивный сброс пытается эксплуатировать большой словарь как можно дольше.

Другое наблюдение состоит в том, что многие файлы (особенно архивы TAR (архивы системы UNIX), содержащие различные виды файлов внутри архива) содержат области с различными типами данных. В процессе работы алгоритма LZW формируется словарь, специально приспособленный для определенного типа данных. При значительном изменении типа данных словарь уже не будет давать хорошего сжатия. Контролируя эффективность сжатия, программа compress может сбрасывать словарь в момент падения производительности.

Другие программы, использующие алгоритм LZW фактически, работают с двумя зарезервированными кодами. В дополнение к коду сброса, они используют специальный код, с помощью которого помечается конец сжатых данных.

Алгоритмы, использованные в программе ZIP

В известном архиваторе ZIP используется множество алгоритмов. Теоретически, программа ZIP могла бы поочередно испытать каждый из этих методов и выбрать наиболее подходящий для данного файла. Но на практике этот подход никогда не используется, так как требует слишком больших затрат времени. Обычно для большинства типов файлов лучшим оказывается самый новый метод. В результате, большинство программ ZIP используют лишь самый новый метод (при желании можно использовать другой метод, воспользовавшись для этого переключателями командной строки). Если сжатия не произошло, и размер файла вырос, то они возвращаются к несжатому файлу.

Но для нас представляют интерес все алгоритмы, использованные в ZIP. Начнем с более старых алгоритмов, редко используемых сегодня, и закончим новыми. Используемый ныне алгоритм Deflation, был принят другими утилитами сжатия (включая GZIP и графический формат PNG) в значительной степени из-за того, что он не имеет никаких патентных ограничений.

Shrinking. Shrinking - модифицированная версия использованного в программе compress алгоритма LZW. Что же в ней появилось нового?

1. Shrinking осуществляет лишь частичный сброс (partial reset) словаря. Вместо того чтобы полностью удалять словарь. Shrinking отбрасывает лишь самые длинные строки. Когда я рассказывал о работе compress, то обращал Ваше внимание на то, что сжатие LZW полагается на существование длинных строк в словаре. Так как после сброса в словаре все же остаются некоторые длинные строки, можно рассчитывать на скромное сжатие даже после сброса.

2. Shrinking осуществляет оптимизацию по изменению размера выходных данных compress, переходя к более длинному коду, только когда это для них необходимо. Например, предположим, программа сжатия сделала в словаре 511 записей. Теперь для создания выходных данных с кодом 512 программа compress должна перейти на десятибитные коды. Однако до того момента, когда потребуется такой код, может пройти некоторое время. Shrinking определяет специальную последовательность, которую он выводит только перед переходом к более длинному размеру кода. Программа распаковки переходит к новым размерам кода, только когда встречает эту специальную последовательность.

Reducing. Конечно, так как программа Shrinking опирается на алгоритм LZW, она также попадает под действие патента. Поэтому в ZIP был включен вариант незапатентованного алгоритма сжатия LZ77. Вместо того чтобы для каждой идентифицированной последовательности выводить код, LZ77 выводит смещение, на которое надо вернуться, и длину повторяющихся данных. Например, abcdabc мог бы быть сжат как abcd, за которым бы следовало смещение 4 и длина 3. То есть чтобы воспроизвести целиком эту последовательность, надо возвратиться на четыре байта и скопировать три байта.

Алгоритмы Reducing, Imploding и Deflation различаются способом хранения смещения и длины в выходных данных и тем, какой тип дополнительного сжатия они используют.

Reducing хранит смещения и длины, используя управляющий код (escape code). Кодировщик записывает каждый несжатый байт, как он есть, перед парой смещение/длина, предваряя его байтом 144 (ASCII DLE). Смещение и длина занимают вместе два или три байта, при этом между собой они делят их по-разному, в зависимости

от значения "фактора сжатия" (compression factors). При более высоких значениях фактора сжатия можно использовать большие смещения, в то время как при меньших значениях можно увеличивать значение длины.

После этой стадии сжатия LZ77 результат сжимается снова простым вероятностным методом. Этот метод формирует список, в котором для каждого значения байта указано, какие байты следуют за ним чаще всего. Например, за символом t в английском языке чаще всего следуют символы h и o. Reducing для хранения наиболее часто встречающихся "сопровождающих" байт использует меньшее количество бит, чем для более редких. Таблица наиболее общих сопровождающих символов присоединяется в начало перед сжатыми данными.

Imploding. Двухступенчатое сжатие, используемое программами Reducing, Imploding и Deflation приследует две цели. Во-первых, можно использовать преимущества сразу двух различных методов сжатия (LZ77 и сжатие Хаффмана). Во-вторых, такой подход позволяет оптимизировать выходные данные LZ77 с помощью использования коротких кодов для наиболее часто встречающихся смещений и длин.

Imploding начинает сжатие LZ77 с ограничения на смещения либо 4096, либо 8192, после чего выбор кодов различной битовой длины для литералов (несжатых последовательностей), смещений и длин осуществляется с помощью сжатия Шеннона-Фано. Три частотных дерева Шеннона-Фано строятся отдельно; после сжатия выходные данные содержат перед каждым литералом или смещением дополнительный бит, который указывает на то, кодирует ли следующий бит литерал или пару смещение/длина.

Deflation. Алгоритм Deflation обрабатывает сжатие LZ77, сохраняя таблицу всех трехбайтовых последовательностей, которые появляются в данных. Если три байта соответствуют записи таблицы, программа обращается к предыдущим данным и смотрит, не распространяется ли это соответствие больше, чем на три байта. Параметр управляет тем, как долго алгоритм будет пытаться найти лучшую последовательность. Установка "Быстрейшее" по существу означает тот факт, что программа сжатия всегда будет использовать первый найденный ею повтор. Установка "Лучшая" инструктирует программу сжатия рассматривать каждое соответствие и найти то, которое лучше всего работает. Программа распаковки должна предоставлять доступ к 32 килобайтам декодированных данных, так как смещения больше 32 К программа сжатия никогда не использует.

После завершения сжатия LZ77 результат представляется в блоках по 32 килобайта и трех встроенных наборов кодов Хаффмана (Huffman). Первый набор кодирует вместе литералы и смещения (таким образом, отпадает необходимость в дополнительном бите, использующимся в алгоритме Imploding), второй кодирует значения длины. Третий набор кодов Хаффмана используется для сжатия первых двух деревьев Хаффмана. Программа распаковки читает первое дерево из начала сжатого блока и использует его для декодирования дерева Хаффмана литерал/смещение и дерева Хаффмана длины. Последние используются для декодирования собственно сжатых данных.

В целом алгоритм Deflation обычно немного лучше с точки зрения сжатия, но немного медленнее алгоритма LZW, используемого программой compress.

Обычно объединение двух или более методов сжатия дает очень небольшой выигрыш. Однако комбинация алгоритмов LZ77 / Хаффман работает неплохо. Это связано с двумя причинами. Во-первых, эти алгоритмы сжимают различные типы данных. LZ77 сжимает данные, имеющие повторяющиеся последовательности байтов, в то время как алгоритм Хаффман сжимает данные, имеющие неравномерное распределение значений байтов. В результате комбинация этих методов способна сжать более широкий круг данных, чем любой из входящих в нее алгоритмов по отдельности. Хотя более важно, что комбинация этих алгоритмов приводит к терпимым результатам даже при самом плохом раскладе. Вероятность того, что в результате неудачного сжатия размер файлов увеличится, крайне мала, так как зачастую увеличение файла из-за плохой работы одного алгоритма будет скомпенсировано эффективной работой другого.

Контрольные примеры

В данном разделе будет рассмотрен контрольный пример архивации и разархивации данных. Примеры даны применительно к программе, они отражают суть происходящих при архивации/разархивации явлений. Следует заметить, что в программе приведён несколько модифицированный алгоритм архивации. От первоначального его отличает то, что код сброса 256 в модифицированном методе не используется. Это упрощение никак не повлияет на работу алгоритма. Дело в том, что код 256 используется в LZ89 для сброса словаря только в достаточно больших архивируемых данных, а в данной программе, эффект от использования сброса кодом 256 не будет виден.

Архивирование данных методом LZ78

Определим для ясности некоторые понятия:

Буфер - некоторая логическая область неограниченного размера, в которую можно добавлять символы.

Последовательность - набор символов от 1 до N, которые следуют друг за другом.

Код - код добавляемой в словарь последовательности. Всегда первым кодом идёт число 257.

Фраза - последовательность символов, которую необходимо заархивировать.

Рассмотрим алгоритм метода.

Дано: фраза, буфер. В программе для упрощения операций над архивом предусмотрен словарь.

1. Смотрим фразу выбираем следующий
символ (первый, при первом проходе)
2. Добавляем символ в буфер
3. Смотрим в стандартном словаре и в словаре пользователя: встречается ли там последовательность из буфера.
- Если да то 1.
- Если нет то 4.
4. -Добавляем последовательность в словарь пользователя
- Переносим из буфера последний символ в логическую
область Q.
- Добавляем в архив код той последовательности,
которая сейчас находится в буфере
- Очищаем буфер
- Буфер = Q
5.Переходим к шагу 1.

Пример 1. Архивация данных

PASTPARASTIC

Выбираем символ "P" (и сразу заносим в буфер, на практике - это бумажка на которой исследуется содержание буфера). Смотрим в словаре. Этот символ уже есть (все ASCII-символы уже педусмотренны в программе - для этого предназначен стандартный словарь). Смотрим следующий символ - "A" (добавляем в буфер). Сочетания "PA" ни в словаре пользователя, ни в стандартном словаре нет, поэтому добавляем её в словарь (пользователя) с кодом 257, добавляем в архив символ P (с кодом 80) и оставляем в буфере только символ "A".

Далее добавляем символ "S". У нас получилась последовательность "AS". Добавляем её в словарь с кодом 258. В архив "A" (код 65). В буфере оставляем "S".

Добавляем "T". В буфере "ST". Добавляем в словарь "ST" под кодом 259. В архив "S" (код 83).В архив В буфере "T".

Добавляем "P". В буфере "TP". Добавляем в словарь "TP" под кодом 260. В архив "T" (код 84). В буфере "P".

Добавляем "A". В буфере "PA". В словаре пользователя уже есть PA. Поэтому опять добавляем символ (следующий). Добавляем "R". В буфере "PAR". Добавляем в словарь "PAR" под кодом 261. В архив "PA" (код 257). В буфере "R".

Добавляем "A". В буфере "RA". Добавляем в словарь "RA" под кодом 262. В архив "R" (код 82). В буфере "A".

Добавляем "S". В буфере "AS". В словаре "AS" уже встречается. Смотрим следующий символ из фразы.

Добавляем "T". В буфере "AST". В словарях "AST" не встречается. Добавляем в словарь "AST" (код 263). В архив идёт "AS" под кодом 258. В буфере "T".

Добавляем "I". В буфере "TI". В словарях нет "TI". Добавляем в словарь "TI" под кодом 264. В архив идёт 84 - код "T". В буфере остаётся "I".

Добавляем "C". В буфере "IC". В словарях нет "IC". Добавляем в словарь "IC" под кодом 265. В архив идёт 73 - код "I". В буфере остаётся "C".

Так как в фразе больше нет символов. Добавляем в архив "C" и выходим из алгоритма.

У нас получился следующий словарь, см. Таблица 3.

Таблица 3

Словарь

PA	257
AS	258
ST	259
TP	260
PAR	261
RA	262
AST	263
TI	264
IC	265

И следующий архив: 80 - 65 - 83 - 84 - 257 - 82 - 258 - 84 - 73 - 67

Пример 2. Разархивация данных.

Разархивация выполняется несколько сложнее нежели архивация. Следует внимательно вникнуть в суть происходящих явлений. При разархивации данных используются те же ключевые понятия, что и при архивации. Хотя есть новое ключевое слово:

Элемент - число из архива

Алгоритм разархивирования данных.

1. Выбираем из фразы следующий элемент последовательности. Переводим этот элемент в символьное представление и добавляем в буфер.
2. Если добавленный элемент имеет код >=257, то формируем буфер-Q, который состоит из последовательности буфера без последнего символа (но не элемента!). Если код добавленного элемента <= 256, то формируем буфер-Q = буфер
3. Смотрим: встречается ли последовательность из буфера-Q в словарях?
 - Если да, то переходим к шагу 1.
 - Если нет, то переходим к шагу 4.
4. Добавляем последовательность из буфера-Q в словарь. Теперь в буфере должна находиться только последовательность, составленная новым элементом, без старого. Добавляем в str (местодержание разархивированной фразы) Буфер-Q за исключением первого символа.
5. Переходим к шагу 1.

Пример 3. Архивация данных, см. таблица 5 .

80 - 65 - 83 - 84 - 257 - 82 - 258 - 84 - 73 – 67

Таблица 5

Словарь

Шаг	Буфер	Буфер-Q	В словарь
1	P	P	
2	PA	PA	PA - 257
3	AS	AS	AS - 258
4	ST	ST	ST - 259
5	TPA	TP	TP - 259
6	PAR	PAR	PAR - 261
7	RAS	RA	RA - 262
8	AST	AST	AST - 263
9	TI	TI	TI - 264
10	IC	IC	IC - 265

Таким образом на выходе мы получили уже знакомую нам фразу "***PASTPARASTIC***".

Алгоритм Хаффмана

Сжимая файл по алгоритму Хаффмана первое что мы должны сделать – это необходимо прочитать файл полностью и подсчитать сколько раз встречается каждый символ из расширенного набора ASCII. Если мы будем учитывать все 256 символов, то для нас не будет разницы в сжатии текстового и EXE файла. После подсчета частоты вхождения каждого символа, необходимо просмотреть таблицу кодов ASCII и сформировать мнимую компоновку между кодами по убыванию. То есть не меняя местонахождение каждого символа из таблицы в памяти отсортировать таблицу ссылок на них по убыванию. Каждую ссылку из последней таблицы назовем "узлом". В дальнейшем (в дереве) мы будем позже размещать указатели которые будут указывает на этот "узел". Для ясности давайте рассмотрим пример:

Мы имеем файл длинной в 100 байт и имеющий 6 различных символов в себе. Мы подсчитали вхождение каждого из символов в файл и получили следующее, см. таблица 6.

Таблица 6

Число вхождений символов в файл

символ	A	B	C	D	E	F
число вхождений	10	20	30	5	25	10

Теперь мы берем эти числа и будем называть их частотой вхождения для каждого символа. Разместим в таблице 7.

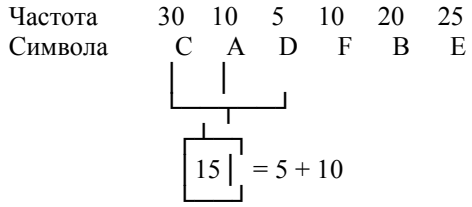
Таблица 7

Частота вхождения символов

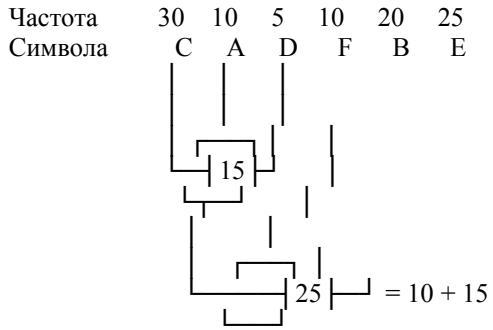
символ	C	E	B	F	A	D
число вхождений	30	25	20	10	10	5

Мы возьмем из последней таблицы символы с наименьшей частотой. В нашем случае это D (5) и какой либо символ из F или A (10), можно взять любой из них

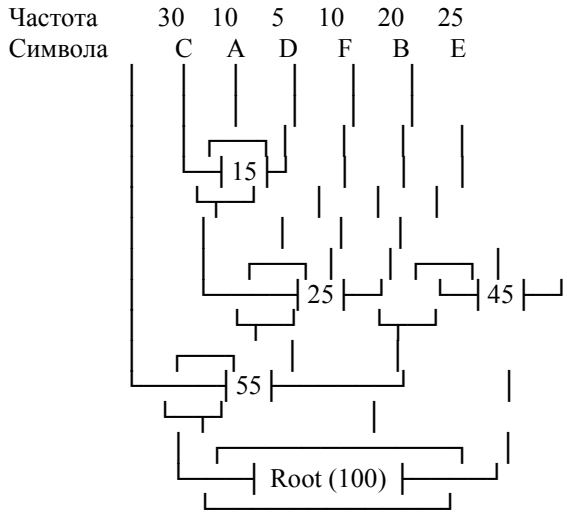
например A. Сформируем из "узлов" D и A новый "узел", частота вхождения для которого будет равна сумме частот D и A :



Номер в рамке - сумма частот символов D и A. Теперь мы снова ищем два символа с самыми низкими частотами вхождения. Исключая из просмотра D и A и рассматривая вместо них новый "узел" с суммарной частотой вхождения. Самая низкая частота теперь у F и нового "узла". Снова сделаем операцию слияния узлов :



Рассматриваем таблицу снова для следующих двух символов (B и E). Мы продолжаем в этот режим пока все "дерево" не сформировано, т.е. пока все не сведется к одному узлу.



Теперь, когда наше дерево создано, мы можем кодировать файл. Мы должны всегда начинать из корня (Root). Кодируя первый символ (лист дерева C) мы прослеживаем вверх по дереву все повороты ветвей и если мы делаем левый поворот, то запоминаем 0-й бит, и аналогично 1-й бит для правого поворота.

Так для C, мы будем идти влево к 55 (и запомним 0), затем снова влево (0) к самому символу. Код Хаффмана для нашего символа C - 00. Для следующего символа (

A) у нас получается - лево, право, лево, лево, что выливается в последовательность 0100. Выполнив выше сказанное для всех символов получим:

C = 00 (2 бита)

A = 0100 (4 бита)

D = 0101 (4 бита)

F = 011 (3 бита)

B = 10 (2 бита)

E = 11 (2 бита)

Каждый символ изначально представлялся 8-ю битами (один байт), и так как мы уменьшили число битов необходимых для представления каждого символа, мы следовательно уменьшили размер выходного файла. Сжатие складывается следующим образом, см. таблицу 8.

Таблица 8

Результаты сжатия

Частота	**первоначально**	**уплотненные биты**	**уменьшено на**
C 30	30 x 8 = 240	30 x 2 = 60	180
A 10	10 x 8 = 80	10 x 3 = 30	50
D 5	5 x 8 = 40	5 x 4 = 20	20
F 10	10 x 8 = 80	10 x 4 = 40	40
B 20	20 x 8 = 160	20 x 2 = 40	120
E 25	25 x 8 = 200	25 x 2 = 50	150

Первоначальный размер файла : 100 байт - 800 бит;

Размер сжатого файла : 30 байт - 240 бит;

240 - 30% из 800 , так что мы сжали этот файл на 70%.

Все это довольно хорошо, но неприятность находится в том факте, что для восстановления первоначального файла, мы должны иметь декодирующее дерево, так как деревья будут различны для разных файлов. Следовательно мы должны сохранять дерево вместе с файлом. Это превращается в итоге в увеличение размеров выходного файла .

В нашей методике сжатия и каждом узле находятся 4 байта указателя, по этому, полная таблица для 256 байт будет приблизительно 1 Кбайт длинной.

Таблица в нашем примере имеет 5 узлов плюс 6 вершин (где и находятся наши символы), всего 11. 4 байта 11 раз - 44. Если мы добавим после небольшое количество байтов для сохранения места узла и некоторую другую статистику - наша таблица будет приблизительно 50 байтов длинны. Добавив к 30 байтам сжатой информации, 50 байтов таблицы получаем, что общая длинна архивного файла вырастет до 80 байт. Учитывая, что первоначальная длинна файла в рассматриваемом примере была 100 байт – мы получили 20% сжатие информации.

Не плохо. То что мы действительно выполнили - трансляция символьного ASCII набора в наш новый набор требующий меньшее количество знаков по сравнению с стандартным.

Что мы можем получить на этом пути ?

Рассмотрим максимум, который мы можем получить для различных разрядных комбинаций в оптимальном дереве, которое является несимметричным.

Мы получим что можно иметь только :

4 - 2 разрядных кода;

8 - 3 разрядных кодов;

16 - 4 разрядных кодов;

32 - 5 разрядных кодов;

64 - 6 разрядных кодов;

128 - 7 разрядных кодов;

Необходимо еще два 8 разрядных кода.

4 - 2 разрядных кода;

8 - 3 разрядных кодов;

16 - 4 разрядных кодов;

32 - 5 разрядных кодов;

64 - 6 разрядных кодов;

128 - 7 разрядных кодов;

254

Итак, имеем итог из 256 различных комбинаций которыми можно кодировать байт. Из этих комбинаций лишь 2 по длине равны 8 битам. Если мы сложим число битов которые это представляет, то в итоге получим 1554 бит или 195 байтов. Так в максимуме, мы сжали 256 байт к 195 или 33%, таким образом максимально идеализированный Huffman может достигать сжатия в 33% когда используется на уровне байта .

Все эти подсчеты производились для не префиксных кодов Хаффмана т.е. кодов, которые нельзя идентифицировать однозначно.

Например, код A - 01011 и код B - 0101. Если мы будем получать эти коды побитно, то получив биты 0101 мы не сможем сказать какой код мы получили A или B, так как следующий бит может быть как началом следующего кода, так и продолжением предыдущего.

Необходимо добавить, что ключом к построению префиксных кодов служит обычное бинарное дерево и если внимательно рассмотреть предыдущий пример с построением дерева, можно убедится, что все получаемые коды там префиксные.

Одно последнее примечание - алгоритм Хаффмана требует читать входной файл дважды, один раз считая частоты вхождения символов, другой раз производя непосредственно кодирование.

Лабораторная работа 1. Метод сжатия LZ78

Задание

Ознакомиться с теоретической частью.

В соответствии с полученным от преподавателя вариантом:

1. Проделать сжатие последовательности символов методом Лемпела-Зива с использованием программы «LZ78».
2. Произвести восстановление символьной последовательности по словарю, с использованием программы «LZ78» .

Использование программы

Программа LZ78 написана в ДВГТУ, на кафедре ИСУ, для организации изучения студентами методов сжатия. В качестве метода сжатия выбран алгоритм Лемпела-Зива, предложенный в 1978 году.

Начало работы с программой

При запуске программы перед вами появится следующее окно, см. рис.1.

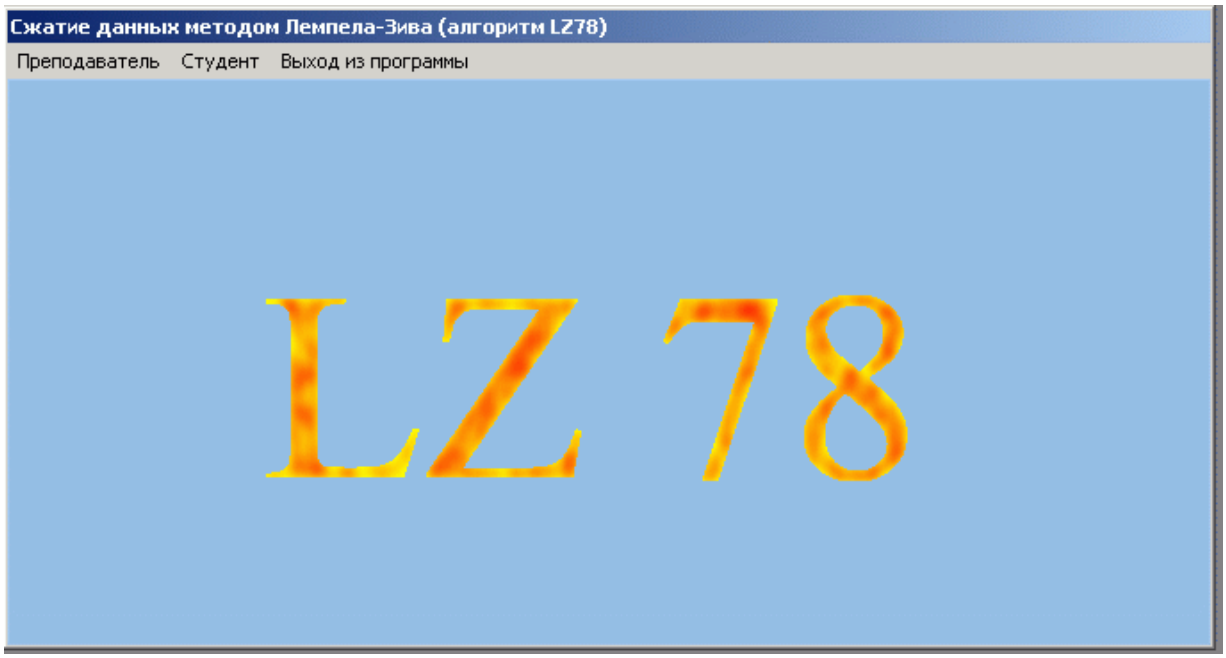


Рис.1 Основное окно программы

Часть №1

Далее Вам необходимо перейти к первой части Вашей работы. Выберете пункт меню "Студент->Часть №1". Программа сразу изменит вид, см. рис.2.

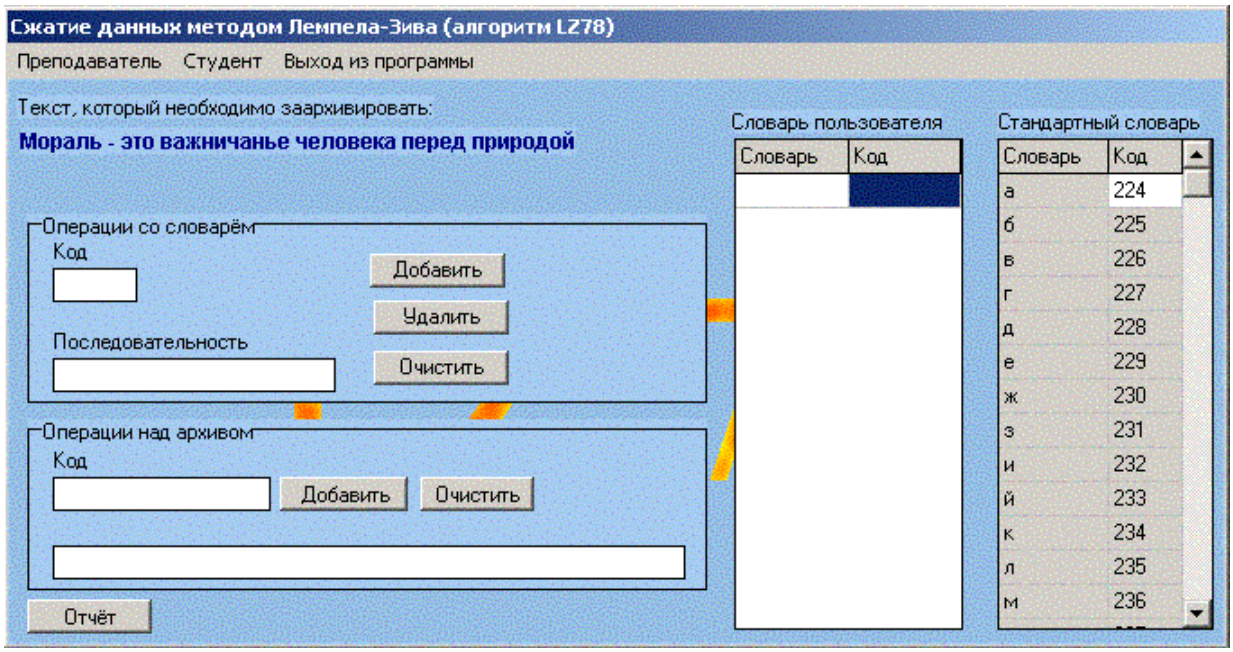


Рис.2 Работа по составлению словаря пользователя

В первой части работы Вам предстоит заархивировать фразу. Фраза эта написана в левом верхнем углу программы.

В нашем примере это "Мораль - это важничанье человека перед природой". Справа вы увидите две области называемые "Словарь пользователя" и "Стандартный словарь".

Стандартный словарь - это ASCII-таблица символов. Этот словарь статичен и его нельзя изменять.

Словарь пользователя - специальная область, для хранения последовательностей lz-алгоритма.

Что бы добавить последовательность в словарь пользователя существует область в программе, которая называется "Работа со словарём".

В поле "Код" Вы должны будете ввести код добавляемой последовательности (порядок построения кодов показан в разделах "АЛГОРИТМЫ" и "КОНТРОЛЬНЫЙ ПРИМЕР").

В поле последовательность - сама последовательность (набирается с клавиатуры). После ввода необходимо нажать добавить и запись встанет в конец словаря пользователя, см. рис.3.

Рис.3 Просмотр добавленных последовательностей в словарь пользователя

Что бы удалить запись из словаря необходимо ввести в поле "Код" код удаляемой записи и нажать кнопку удалить. Для очистки всего списка необходимо нажать кнопку "Очистить" в области "Работа со словарём".

Запомните! Порядок последовательностей в словаре имеет значение.

Следующим этапом является заполнение архива (архивирование информации) с использованием словарей. То, как происходит архивирование показано в разделе "Контрольный пример".

За работу с архивом отвечает область "Операции над архивом". В поле "Код" введите код, (из словаря пользователя или стандартного словаря) и нажмите "Добавить".

Запись попадёт в поле архива в специальном формате.

Для очистки поля архива нажмите "Очистить".

После выполнения первой части задания необходимо сохранить результаты архивирования нажав на кнопку "Отчёт".

Далее следует перейти к следующей части работы выбрав меню "Студент->Часть №2"

Часть №2

Внешне существует два отличия от первой части программы.

Во первых, вместо фразы, которую необходимо было заархивировать теперь появилась последовательность, которую необходимо разархивировать.

В нашем примере это "211-236-32-226-241-229-227-228-224-259-32-228-243-240-224-234-224-245-32-243-32-261-240-228-246-224-".

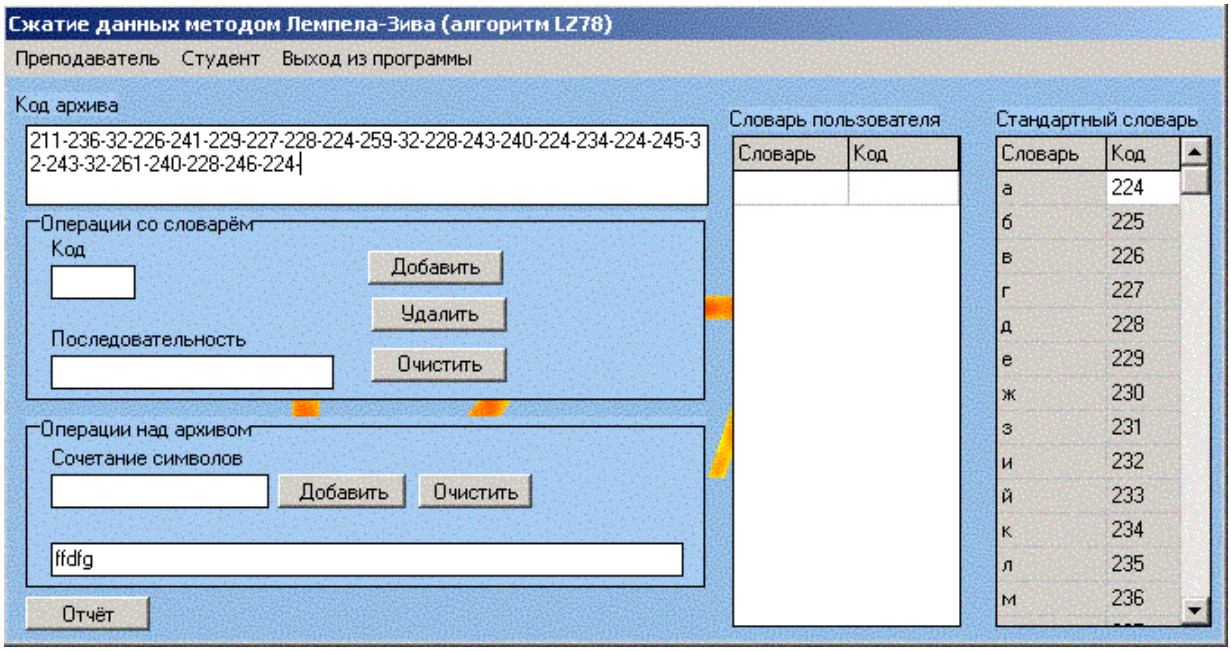


Рис.4. Вторая часть работы - разархивация

Во вторых в области "Операции над архивом" вместо добавления кодовых последовательностей в виде цифр.

Вы добавляете последовательности разархивированных символов. Внизу в поле архива, вы увидите полную фразу, которую вы получите в результате разархивации.

После заполнения словаря и заполнения фразы в области "Операции над словарём" Вам необходимо будет сохранить отчёт о проделанной работе. Для этого нажмите кнопку "Отчёт" и сохраните отчёт под выбранным именем.

Не забывайте, что у первой и второй части заданий разные отчёты. *Помните! Даже если вы угадаете фразу, вы должны будете заполнить словарь пользователя, без правильного заполнения которого невозможно будет составить правильный отчёт.*

Лабораторная работа № 2. Метод сжатия Хаффмана

Задание

Ознакомиться с теоретической частью.

В соответствии с полученным от преподавателя вариантом необходимо построить бинарное дерево сжатия и правильно заполнить все поля промежуточных результатов.

Использование программы

На рис.5 показано основное окно программы «Сжатие данных»

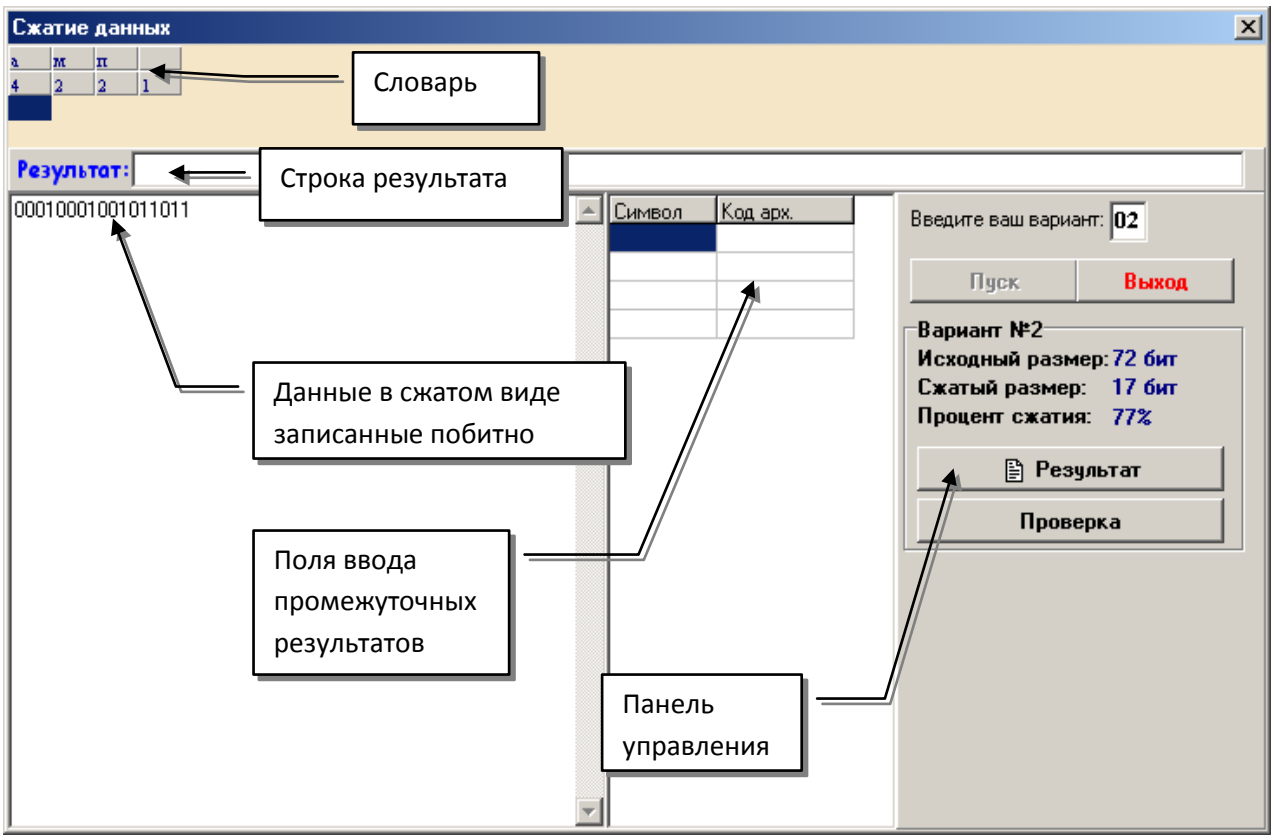


Рис. 5 Основное окно программы

Ход работы:

1. Введите номер варианта в соответствующем поле, расположенном над панелью управления.
2. Нажмите кнопку “Пуск”.
3. На экране появятся следующие данные:
    a. *Словарь* – на основании него вам необходимо построить бинарное дерево в соответствии, с которым сжато исходное предложение. Словарь содержит все символы, встречающиеся в строке и их частоту. Словарь отсортирован по частоте по убыванию.
    b. *Сжатые данные* – строка соответствующая вашему варианту в жатом виде из которого при правильном заполнении полей ввод промежуточных результатов будет восстановлена исходная строка.
    c. *Поля ввода промежуточных результатов* – в данные поля необходимо после построения бинарного дерева сжатия внести сами символы и их коды в сжатом виде. На экран выводится ровно столько пар полей сколько их в словаре. При заполнении они все должны быть заполнены и не должно остаться лишних символов.
4. Построение бинарного дерева сжатия – построение дерева сжатия производится по методу Хаффмана, при этом на алгоритм накладывается следующее условие: при выборе минимального элемента если встречается более одного узла с одинаковой частотой ***необходимо выбирать самый правый из таки узлов.***
5. После того как дерево построено, заполните поля промежуточных результатов.
6. Нажмите кнопку “Результат”.
7. После того как все действия будут выполнены правильно, при нажатии кнопки «результат» на панели управления, в строке результата будет выведена строка соответствующая вашему варианту. Иначе, так как строка формируется из сжатой информации, при неправильном заполнении полей промежуточных результатов строка будет восстановлена не верно.

Требования к отчету

1. Краткие теоретические сведения

2. Задание
3. Результаты выполнения вместе с расчетами.
4. Обобщенные выводы по результатам работы.

Литература

1. Jacob Ziv, Abraham Lempel. A Universal Algorithm for Sequential Data Compression IEEE Transactions on Information Theory, 23(3), pp. 337–343, May 1977.
2. http://www.intuit.ru/studies/courses/2256/140/lecture/2059?page=2 Владимир Лидовский, Описание алгоритма LZ78 в курсе лекций "Основы теории информации и криптографии" // Интуит.ру, 11.04.2007
5. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

Кафедра информационных систем управления

С. С. Пашин

РАСЧЁТ МАРШРУТОВ В СЕТЯХ

Методические указания к лабораторной работе

по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток

2013

Приведены основные сведения о методах построения маршрутов и матриц маршрутов при передаче сообщений по сетям телеобработки данных с коммутацией сообщений и каналов Цель указаний - получение практических навыков проектирования и расчёта сетей телеобработки данных.

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

Введение

Целью настоящей работы является ознакомление с общими методами расчёта маршрутов при передаче информации в сетях телеобработки данных с коммутацией сообщении и каналов, получение практических навыков проектирования и расчёта сетей телеобработки данных.

1. Методические указания

1.1. Кратчайшие пути в сетях связи

Распределение информации по сетям связи производится с учётом оптимальности путей, по которым передаётся информация. При этом, очевидно, информацию целесообразно передавать в первую очередь по наиболее коротким или, как говорят, кратчайшим путям.

Для оценки «длины» пути могут быть использованы различные критерии. Например, число транзитных узлов в пути, протяжённость пути (в километрах), качество тракта, вероятность передачи информации, вероятность прерывания связи, величина затухания и т.п. Значения многих из этих критериев не являются независимыми, поэтому рассматриваются лишь некоторые из них. В настоящее время основным критерием длины пути часто принимают критерий числа в нём транзитных узлов или ветвей. В ряде случаев в качестве дополнительного критерия рассматривается протяжённость пути.

Кратчайшим путём передачи информации считается тот. у которого длина имеет наименьшее значение по сравнению с другими. Все способы выбора кратчайших путей основаны на утверждении, что если кратчайший путь $М_{in}$ от произвольного узла $УЗ_i$ к узлу $УЗ_n$ проходит через промежуточные узлы $УЗ_p$, ... ,$УЗ_k$ (рис. 1.1), то кратчайшие пути $М_{pn}$ $М_{kn}$ от узлов $УЗ_p$, ... ,$УЗ_k$ к узлу $УЗ_n$ соответственно являются частями кратчайшего пути $М_{in}$ от $УЗ_i$ к $УЗ_n$

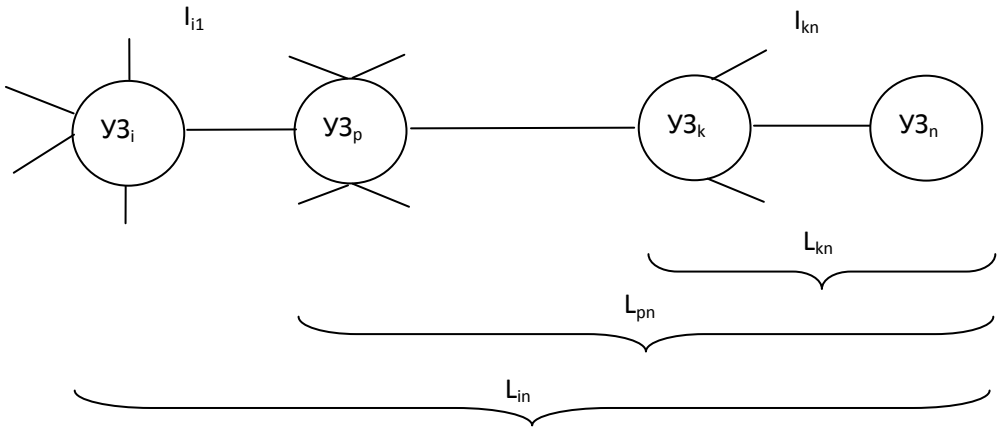


Рис. 1.1. Кратчайшие пути передачи информации

Если длина пути $M_{pn} = L_{pn}$, то

$$L_{in} = I_{ip} + I_{pn}.$$

Так как путь M_{in} является кратчайшим.

$L_{in} = \min(I_{ij} + L_{jn})$,

где j = 1, ... ,N; N - число узлов сети.

Таким образом, чтобы найти кратчайший путь от узла i к узлу N. необходимо просмотреть все возможности пути и выбрать из них путь с наименьшей длиной.

Для определения этой длины известны две группы способов:

1. нумерации узлов и ветвей;
2. матричные.

Один из наиболее характерных для первой группы способов определения длины кратчайших путей состоит в выполнении следующих операций:

выделенному узлу N приписывается вес (т.е. расстояние от рассматриваемого узла до узла N) $W_n = 0$, а каждому из остальных узлов сети

$W_i = \infty$;

выбирается произвольный узел i (i=l, ... ,N; i≠N) и проверяется неравенство

$$W_i > (I_{ij} + Wj), \qquad (1.1)$$

где Wj - вес соседнего узла.

Если неравенство (1.1) выполняется, то прежний вес W_i узла i заменяется весом

$$W_i = I_{ij} + W_j;$$

в противном случае вес узла остаётся без изменений. Указанная процедура перерасчёта узлов производится до тех пор, пока хотя бы для одного узла выполняется неравенство (1.1). Веса W_i узлов после перерасчёта будут равны длине кратчайшего пути от узла i к выделенному узлу N. Повторяя эту процедуру для каждого из N узлов сети, можно рассчитать кратчайшие пути между всеми узлами сети

Матричный способ определения кратчайших путей, позволяющий одновременно найти длину кратчайших путей между всеми узлами сети, основывается на применении операций над матрицами расстояний.

Структуру сети связи с указанием длины её ветвей можно описать в виде матрицы длин ветвей сети

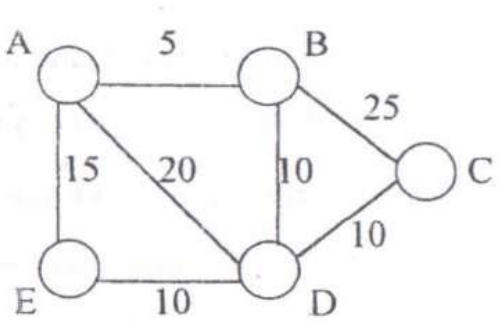
$$L^1 : L^1 = \left\| I_{ij}^1 \right\|,$$

где элемент I^1_{ij} указывает длину ветви B_{ij}. В матрице длины ветвей элемент главной диагонали всегда равен нулю. т.к. принимается, что расстояние внутри узла равно нулю. Если между парой узлов отсутствует связь, то соответствующий элемент принимается равным ∞.

Очевидно, матрица ветвей неориентированной сети всегда симметрична относительно своей главной диагонали; для ориентированной сети связи она может быть несимметричной.

Пример

Если сеть задана в виде графа связи, изображённого на рис. 1.2, цифры на ветвях которого соответствуют длинам ветвей, то матрица длин ветвей L^1 будет иметь вид



$L^1=$

	A	B	C	D	E
A	0	5	∞	20	15
B	5	0	25	10	∞
C	∞	25	0	10	∞
D	20	10	10	0	10
E	15	∞	∞	10	0

Рис. 1.2. Граф сети

Возведём теперь матрицу L^1 в квадрат $L^2 = L^1 \cdot L^1$. где элемент I^2_{ij} равен

$$I_{ij}^2 = \sum_{k=1}^{N} (I_{ik}^1 + I_{kj}^1) = I_{i1}^1 \cdot I_{1j}^1 + I_{i2}^1 \cdot I_{2j}^1 + ... + I_{iN}^1 \cdot I_{Nj}^1 \qquad)$$

Интерпретируя теперь умножение как последовательное, а сложение -, как параллельное соединение ветвей, легко понять, что произведение $I^1_{ik} \cdot I^1_{kj}$ соответствует длине двухтранзитного пути (т.е. пути проходящего через две транзитные ветви сети), от узла i к узлу j через узел k (рис. 1.3. а), а сумма,

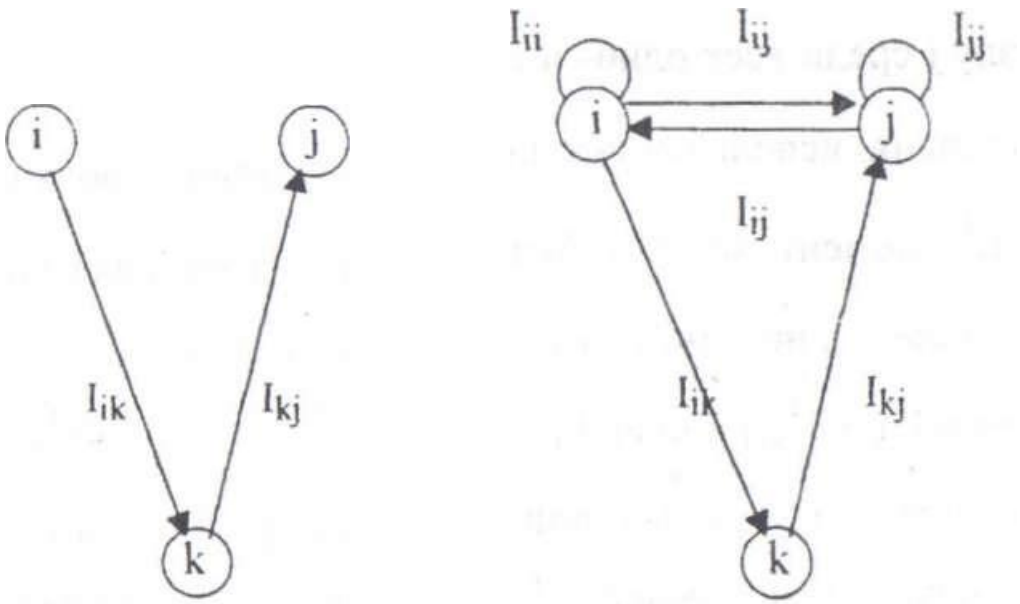


Рис. 1.3. Двухтранзитные пути

например, трёх произведений $I_{ii}^1 \cdot I_{ij}^1 + I_{ij}^1 \cdot I_{jj}^1 + I_{ik}^1 \cdot I_{kj}^1$ - сумме длин двухтранзитных путей (рис. 1.3, б).

Заметим при этом, что произведения $I^1_{ii} \cdot I^1_{ij}$ и $I^1_{ij} \cdot I^1_{jj}$ фактически соответствуют длинам однотранзитных путей (т.е. путей, включающих только одну ветвь), так как длина пути внутри узла (т.е. I^1_{ii} и I^1_{ij}) не принимается во внимание. Поэтому для подсчёта длины каждого из таких транзитных путей необходимо операцию умножения заменить операцией сложения, т.е. вместо $I^1_{ik} \cdot I^1_{jk}$ будем иметь $(I^1_{ik} + I^1_{jk})$.

Если же имеется несколько параллельных одно- и двухтранзитных путей (например, три на рис. 1.3, б), то для определения длины кратчайшего пути между узлами необходимо операцию сложения заменить операцией выбора из всех длин минимальной длины одно- и двухтранзитных путей, т.е. вместо выражения (1.2) будем иметь

$$I_{ij}^2 = \min(I_{ik}^1 + I_{kj}^1) = \min[(I_{i1}^1 + I_{1j}^1);(I_{i2}^1 + I_{2j}^1);\ldots;(I_{iN}^1 + I_{Nj}^1)]$$

k=1…N.

Таким образом, элемент I^2_{ij} матрицы L^2 равен длине кратчайшего пути от узла i к узлу j среди всех одно- и двухтранзитных путей. Возведём матрицу L^1 в r-ю степень, используя указанные выше операции, и получим матрицу $L^r = L^{r-1} \cdot L^1$, элемент которой будет равен длине кратчайшего пути от узла к узлу среди всех одно-, двух- и т.д. r-транзитных путей:

$$I_{ij}^{\mathrm{r}} = \min(I_{ik}^{\mathrm{r}} + I_{kj}^1) = \min[(I_{i1}^{\mathrm{r}} + I_{1j}^1);(I_{i2}^{\mathrm{r}} + I_{2j}^1);\ldots;(I_{iN}^{\mathrm{r}} + I_{Nj}^1)]$$

Легко показать, что при наличии в сети n узлов транзитных ветвей внутри петель не может быть больше n-1. Следовательно, для определения длин кратчайших путей для всех узлов сети среди возможных путей может потребоваться вычисление матрицы L^r, у которой r<n-1. Для конкретной сети может оказаться, что при r < n-1

$$L^r = L^{r-1} \qquad (1.3)$$

При выполнении тождества (1.3) всегда

$$L^{r-1} = L^r,$$

и вычисление матрицы более высокой степени прекращается.

Матрица L^r при выполнении условия (1.3) называется дисперсной и обозначается через D, т.е

$$D = L^{r-1} = L^r = \| d_{ij} \|.$$

Таким образом, элементы дисперсной матрицы равны длинам кратчайших путей между соответствующими узлами сети связи. Поэтому дисперсная матрица иногда называется матрицей кратчайших путей.

Рассмотренные способы позволяют определить длину кратчайшего пути, но не указывают те ветви, которые его составляют. Для определения пути необходимо применить дополнительную процедуру. Так, если применяется способ нумерации узлов, то дополнительная процедура основана на учёте свойства веса узла i, состоящего в том, что существует узел j, $j \neq i$ для которого выполняется равенство

$$W_i = I_{ij} + W_j. \qquad (1.4)$$

Из выражения (1.4) следует, что

$$W_i - W_j = I_{ij} \qquad (1.5)$$

Поэтому если выполняется условие (1.5), то кратчайший путь от узла i проходит через ветвь B_{ij}. Определяя затем ветвь на узле j, для которого выполняется условие (1.5), найдём следующую ветвь, входящую в этот кратчайший путь. Так, шаг за шагом, мы сможем найти все ветви образующие кратчайший путь.

Совокупность путей (от узла i ко всем остальным узлам) сети можно представить в виде дерева путей (рис. 1.4) от узла A сети связи, представленной на рис. 1.2.

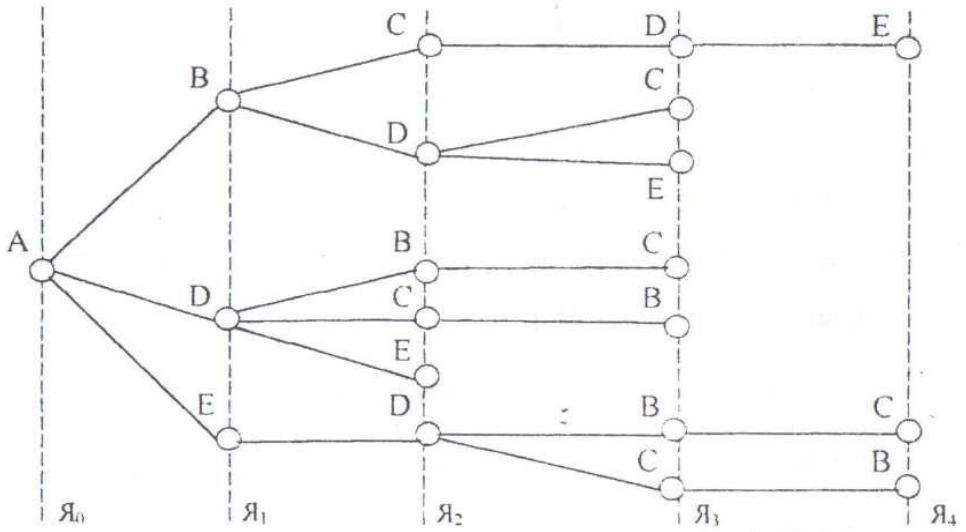


Рис. 1 .4. Дерево путей

С исходящим узлом сети (в нашем случае узлом А) сопоставляется вершина А. называемая корнем в ярусе дерева путей. С вершинами дерева, расположенными в ярусе $Я_{i+1}$ дерева путей, сопоставляются узлы сети, соседние (инцидентные) хотя бы одному из узлов сети, с которыми сопоставлены вершины, расположенные в ярусе $Я_i$ дерева.

Например, в ярусе $Я_1$ дерева (рис. 1.4) расположены вершины B, D, E, т.к. соответствующие им узлы являются соседними узлу А.

1.2. Способы выбора оптимального плана распределения информации на сети

Под планом распределения информации на коммутируемой сети и сети коммутации сообщений понимается заданная очерёдность выбора исходящих направлений из каждого узла ко всем остальным узлам сети. При этом установление очерёдности выбора исходящих направлений зависит от длины рассматриваемых путей, поэтому исходящее направление из рассматриваемого узла связи, которое совпадает с кратчайшим путём, относится к направлению первого выбора и т.д.

Рассмотрим выбор плана распределения информации для коммутируемой сети применительно к сети коммутации каналов. Порядок выбора исходящих направлений (ветвей) $УК_i$ (i-го узла коммутации) ко всем остальным узлам сети. т.е. план распределения информации для узла $УК_i$, можно представить матрицей маршрутов M_i для $УК_i$. В этой матрице число столбцов равно (N-1), где N число узлов на сети (столбец в матрице M_i для узла i не отводится), а число строк - числу n соседних с

рассматриваемым $УК_i$ узлов. Элемент m_{jr} матрицы M_i указывает номер очерёдности выбора ветви B_j при установлении соединения к узлу $УК_r$, т.е. $m_{jr}\{1,...,n\}$.

		$\mathbf{УК_1}$	$\mathbf{УК_2}$	.	.	.	$\mathbf{УК_N}$
	$\mathbf{B_{i,j1}}$	$m_{i,1.1}$	$m_{i,1.2}$	.	.	.	$m_{i,1.N}$
	$\mathbf{B_{i,j2}}$	$m_{i,2.1}$	$m_{i,2.2}$	.	.	.	$m_{i,2.N}$
$\mathbf{M_{i=}}$	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	$\mathbf{B_{i,jn}}$	$m_{i,n.1}$	$m_{i,n.2}$	.	.	.	$m_{i,n.N}$

Все существующие способы получения очерёдности выбора направлений можно разделить на две группы: детерминированные и статистические. В свою очередь эти способы делятся на разовые и групповые.

Разовые детерминированные способы определяют очерёдность выбора направлений обслуживания только для одной заявки на установление соединения. тогда как групповые - для обслуживания группы заявок. При этом матрицы маршрутов вычисляются применительно к ситуации на сети, сложившейся на данный момент без учета предшествующих ситуаций.

Статистические способы позволяют получить рекомендации об очередности выбора исходящих направлений на основе статистики о возможной длине пути по вероятности отказа в том или ином направлении, полученной в результате обслуживания предыдущих заявок. При этом разовые статистические способы дают возможность корректировать матрицу маршрутов после обслуживания каждой заявки, а групповые - после обслуживания нескольких заявок.

Рассмотрим наиболее характерные для этих групп способы получения плана распределения информации; способ рельефов и матричный способ.

1.3. Способ рельефов

Этот способ относится к групповым детерминированным способам, в которых в качестве критерия длины пути используется количество транзитных путей (транзитов).

Представим сеть связи в виде графа, на каждой ветви которого стрелками укажем направление связи. На сети выделим один узел, пусть $УК_A$, по отношению к которому сформулируем так называемый рельеф Стрелками, исходящими из УК, соседних с $УК_A$, припишем цифру 1; всем неотмеченным стрелкам, входящим в УК, от которого отходит хотя бы одна стрелка, отмеченная цифрой 1, припишем цифру 2 и так далее до тех пор, пока не отметим все стрелки. После этого будем говорить, что сформирован А-рельеф (рис. 1.5).

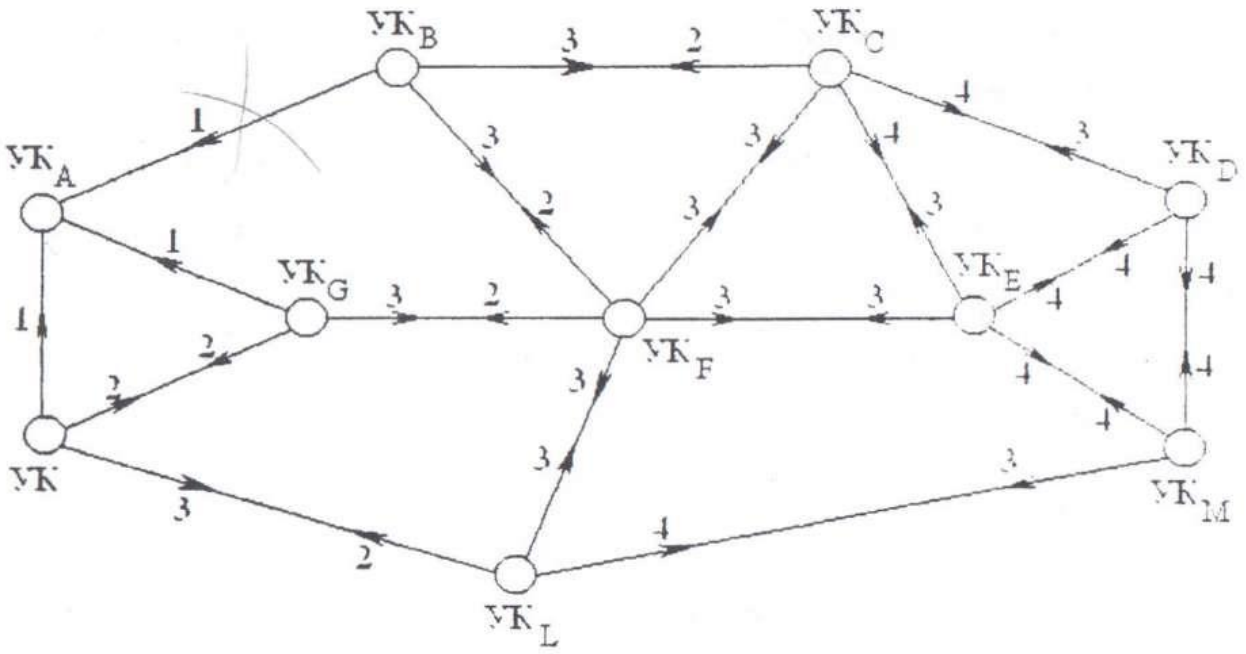


Рис.1.5 А-рельеф

Цифры на стрелках называются А-высотами. При этом стрелкам, исходящим из $УК_A$, цифры (А-высота) А-рельефа не приписываются.

Указанным способом легко сформировать рельеф для каждого УК. Говорят, что ветвь имеет А-высоту S, если соответствующая ей функция в А-рельефе принимает значение S. Таким образом, каждая ветвь имеет (N-1) высот, где N - число узлов коммутации на сети.

Поиск кратчайшего пути по рельефу, например, к $УК_A$ из любого другого УК состоит в отыскании в каждом промежуточном УК ветви с исходящей стрелкой, которой приписано минимальное число. Пусть соединение устанавливается из $УК_E$ в $УК_A$. Тогда

вызов из $УК_E$ направляется по ветви, имеющей минимальную А-высоту. Такой способ обеспечивает установление соединения по оптимальному пути.

Рельефы сети могут быть представлены в виде матриц рельефов. При этом для каждого узла i должна быть составлена своя матрица рельефов вида

$$\mathbf{P_i} = \begin{array}{c|cccc} & \mathbf{УК_1} & . & . & \mathbf{УК_N} \\ \mathbf{B_{i,1}} & r_{i,1.1} & . & . & r_{i,1.N} \\ . & . & . & . & . \\ . & . & . & . & . \\ \mathbf{B_{i,j}} & r_{i,n.1} & . & . & r_{i,n.N} \end{array}$$

в которой элемент $r_{i,k}$ соответствует К-высоте ветви B_j, исходящей из $УК_i$. Например, матрица рельефов для $УК_E$ в которой заполнен один столбец, соответствующий А-рельефу, имеет вид

$$\mathbf{P_E}= \begin{array}{c|ccc} & A & . & . & . \\ \mathbf{B_{EC}} & 3 & . & . & . \\ \mathbf{B_{ED}} & 4 & . & . & . \\ \mathbf{B_{EM}} & 4 & . & . & . \\ \mathbf{B_{EF}} & 3 & . & . & . \end{array}$$

От матриц рельефов можно легко перейти к матрице маршрутов

$$\mathbf{M_E}= \begin{array}{c|ccc} & A & . & . & . \\ \mathbf{B_{EC}} & 1 & . & . & . \\ \mathbf{B_{ED}} & 3 & . & . & . \\ \mathbf{B_{EM}} & 4 & . & . & . \\ \mathbf{B_{EF}} & 2 & . & . & . \end{array}$$

Рассмотрим пример поиска оптимального пути по изменившейся сети Пусть в сети на рис. 1.5 вышла из строя ветвь B_{BA} и соединение устанавливается из $УК_E$ в $УК_A$. Как и ранее сигнал установления пришёл в $УК_B$ по пути

$$M_{EB}= \{УК_E, УК_C, УК_B\},$$

но из $УК_B$ вызов посылается по ветви B_{GA} Итак, сигнал установления соединения до УК идет по пути

$M_{EA} = \{УК_E, УК_C, УК_B, УК_F, УК_G, УК_A\}$, который не является оптимальным. В этом случае говорят, что на сети сформирован неправильный рельеф. Для того чтобы сделать правильный, необходима его коррекция, которая может быть выполнена так же, как и операция его формирования.

В период между коррекциями рельефов для сети может существовать неправильный рельеф. Поэтому те соединения, которые в это время будут устанавливаться, могут проходить не по кратчайшим путям. Выбирая необходимую частоту обновления рельефа, которая для реальной сети находится в пределах 0.5 - 5 мин . можно добиться в среднем достаточно высокой степени оптимизации плана распределения информации. Аналогично метод применяется при коммутации сообщений, только здесь считается, что по сети проходит не сигнал установления соединения, а само сообщение.

1.4. Матричный способ

Как и способом рельефов, матричным способом план распределения информации определяется не для каждого вызова, а для группы вызовов, возникающих в интервале времени между двумя его коррекциями. Матричный способ определения плана распределения информации основан на матричном способе определения длины кратчайшего пути, включая последний в качестве первого этапа.

В начале второго этапа составляется модернизированная матрица длины ветвей сети Г, нулевые элементы главной диагонали которой, в отличие от матриц длин ветвей, имеют значения ∞. По матрице длин ветвей L^1 можно получить модернизированную матрицу длин ветвей Г (для сети 1.2):

Г=

	A	B	C	D	E
A	∞	5	∞	20	15
B	5	∞	25	10	∞
C	∞	25	∞	10	∞
D	20	10	10	∞	10
E	15	∞	∞	10	∞

Замена элемента I_{ii} с нуля на ∞ означает, что длина пути в УК принимается бесконечно большой. Это даёт возможность не рассматривать все пути, проходящие через исходный узел, т.е. позволяет исключить путь (B_{ii}, B_{jj}), изображённый на рис. 1.3, б. Полученная указанным способом модернизированная матрица длин Г = $||\gamma_{ij}||$ умножается на дисперсную матрицу D с использованием тех же операций, что и ранее. При умножении матрицы Г на D образуется матрица Δ = Г·D, элементы которой используются для получения дистанционных матриц (т.е. матриц величин второго и так далее кратчайших путей) я матриц маршрутов. Каждый δ_{ii} элемент матрицы Δ = . δ_{ij} имеет вид

$$\delta_{ij} = MINk[(\gamma_{i1} + d_{1j});(\gamma_{i2} + d_{2j});\ldots;(\gamma_{in} + d_{nj})] \qquad (1.6)$$

В выражении (1.6) MINk означает, что минимум может браться по кратчайшему пути, (k=2 и т.д.).

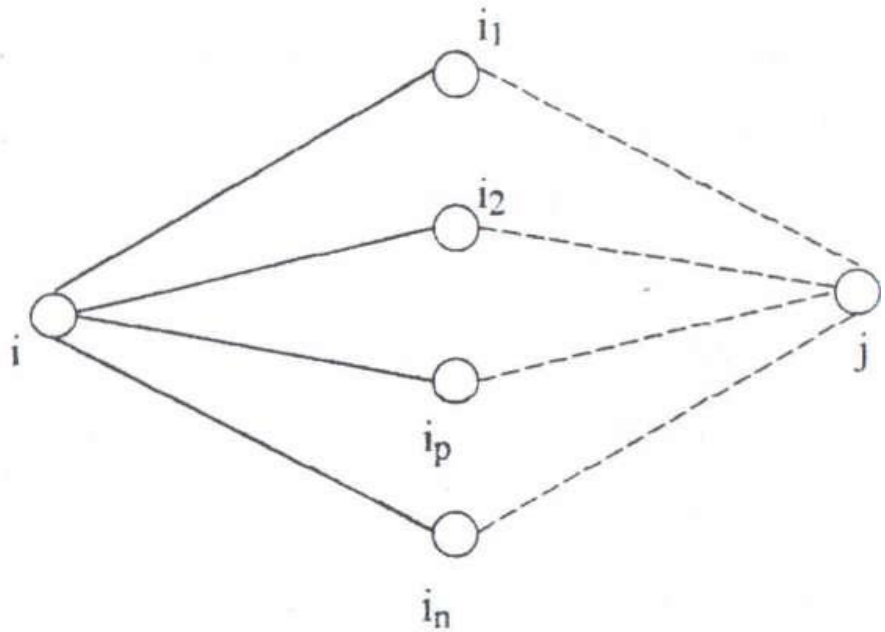


Рис. 1.6. Транзитные узлы

Заметим, что каждый из членов ($\gamma_{ip} + d_{pj}$) выражения (1.6) определяет длину пути от узла i к узлу j, если первым транзитным узлом после узла i на пути к узлу j будет узел P, $P \in \{1,...,N\}$ (рис. 1.6). Легко понять, что если узел P не является соседним узлу i, то член ($\gamma_{ip} + d_{pj}$) равен ∞. В связи с тем, что $\gamma_{ii} = \infty$, член ($\gamma_{ii} + d_{jj}$) не обязательно равен ∞. Таким образом, число членов выражения (1.6), не равных ∞ равно числу n соседних УК (см. рис. 1.6), т.е. числу исходящих из УК направлений (ветвей).

Величина минимального члена выражения (1.6), определяющая длину кратчайшего пути от узла i к узлу j через узел P:

$$\delta_{ij} = MINk[(\gamma_{i1} + d_{1j});(\gamma_{in} + d_{nj});...;(\gamma_{ip} + d_{pj})] \qquad (1.6)$$

заносится в качестве элементов δ_{ij} в так называемую дистанционную матрицу 1 -го выбора $\Delta^1 = \| \delta_{ij} \|$.

Величина второго по значению члена выражения (1.6), определяющего длину второго по протяжённости пути после кратчайшего пути от узла i к узлу j:

$$\delta_{ij}^2 = MIN[(\gamma_{i1} + d_{1j});...;(\gamma_{in} + d_{nj})] \qquad (1.8)$$

заносится в качестве d_{ij} в дистанционную матрицу 2-го выбора $\Delta^2 = \| \delta_{ij} \|$.

При наличии n соседних узлов, очевидно, можно легко получить дистанционные матрицы Δ^1, Δ^2,..., Δ^n. От матрицы Δ легко перейти к тем же матрицам маршрутов для каждого узла.

Для того, чтобы получить матрицу маршрутов для $УК_j$ необходимо найти минимальный член выражения (1.6) при k=p, p=l,2,...,N. Пусть минимальным членом выражения будет член ($\gamma_{ip} + d_{pj}$). Тогда это означает, что k-м по длине путём от $УК_i$, к $УК_j$ будет путь, проходящий через $УК_p$, а ветвь $B_{i,p}$, соединяющая узлы $УК_i$ и $УК_p$, входит в путь k-го выбора. Следовательно, элемент m_{ip} матрицы маршрутов для $УК_i$ равен значению индекса k, т.е. $M_i = B_{ip}[m_{ipk}]$.

Таким образом, можно получить матрицы маршрутов для каждого узла сети, указывающие очерёдность выбора исходящих направлений (ветвей) ко всем другим узлам сети, и дистанционные матрицы, характеризующие длину соответствующих путей.

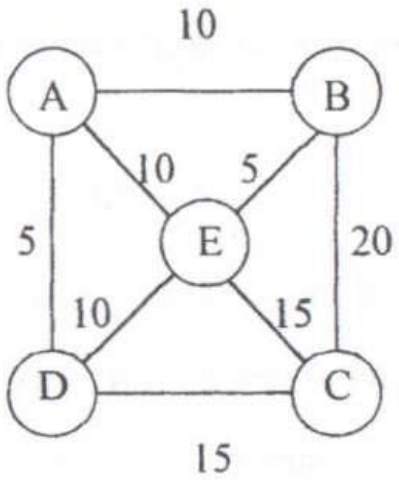
1.5 Пример расчёта сети

Пусть задана сеть телепередачи в виде матрицы, длин ветвей (рис. 1.7,а). Необходимо построить граф связи и определить длину кратчайших путей между всеми парами узлов сети.

L^1=

	A	B	C	D	E
A	0	10	∞	5	10
B	10	0	20	∞	5
C	∞	20	0	15	15
D	5	∞	15	0	10
E	10	5	15	10	0

а)



б)

Рис. 1.7. Матрица и граф длин ветвей

Из первой строчки матрицы определяем, что узел А непосредственно связан с узлами В, D, Е, причём длина соответствующих ветвей 10, 5 и 10. Из второй строчки матрицы определяем, что узел В непосредственно связан с узлами А, С и Е и т.д. Граф связи представлен на рис. 1.7, б.

Далее вычисляем матрицу $L^2 = L^1 \cdot L^1$ Начинаем с элемента

$$I^2_{AA} = \min[(I^1_{AA} + I^1_{AA});(I^1_{AB} + I^1_{BA});(I^1_{AC} + I^1_{CA});(I^1_{AD} + I^1_{AD});$$
$$(I^1_{AE} + I^1_{EA})] = \min[(0+0);(10+10);(\infty+\infty);(5+5);(10+10)] =$$
$$\min[0;20;\infty;10;20] = 0$$

Следует отметить, что элементы главной диагонали матрицы L^1, i=l,2,...,r всегда равны нулю, поэтому их можно не вычислять. Аналогично вычисляем элемент

$$I_{BA}^2 = \min[(0 + 10); (10 + 0); (\infty + 20); (5 + \infty); (10 + 5)] = 10$$

Находя таким же образом и остальные элементы, получим матрицу

L^2=

	A	B	C	D	E
A	0	10	20	5	10
B	10	0	20	15	5
C	20	20	0	15	15
D	5	15	15	0	10
E	10	5	15	10	0

Видно, что $L^2 \neq L^1$. Поэтому вычислим $L^3 = L^2 \cdot L^1$. Элемент I^3_{AA} не вычисляем и сразу принимаем равным нулю. Элемент

$$I_{AB}^2 = \min[(I_{AB}^2 + I_{BB}^1); (I_{AC}^2 + I_{CB}^1); (I_{AD}^2 + I_{DB}^1); (I_{AE}^2 + I_{EB}^1)] = 10$$

Аналогично находим остальные элементы, получим

L^2=

	A	B	C	D	E
A	0	10	20	5	10
B	10	0	20	15	5
C	20	20	0	15	15
D	5	15	15	0	10
E	10	5	15	10	0

=D

Так как $L^3 = L^2$, то $D = L^3$, элементы которой равны длинам кратчайших путей между соответствующими парами узлов сети.

Рассмотрим другой пример. Сеть связи задана графом связи (рис. 1.7, б) и матрицей длин ветвей L^1 (рис. 1.7, а). Необходимо вычислить дистанционные матрицы и построить матрицу маршрутов для $УК_A$. По матрице L^1 составляем модернизированную матрицу

Г=

	A	B	C	D	E
A	∞	10	∞	5	10
B	10	∞	20	∞	5
C	∞	20	∞	15	∞
D	5	∞	15	∞	10
E	10	5	15	10	∞

Затем эту матрицу Г умножаем на дисперсную матрицу, которая была получена в предыдущем примере, т.е. Δ = Г·D. Для этого в начале найдём элемент δ_{AB} матрицы Δ. Заметим при этом, что элементы δ_{ii} очевидно, всегда равны ∞, поэтому мы не будем их вычислять.

$$\delta_{AB} = MINk[(\gamma_{AA} + d_{AB});(\gamma_{AB} + d_{\text{BB}});(\gamma_{AC} + d_{\text{CB}});$$
$$(\gamma_{AD} + d_{\text{DB}});(\gamma_{AE} + d_{\text{EB}})] = \text{MIN}[(\infty + 5);(5+15);(10+15)] =$$
$$\text{MIN}[\infty;10;\infty 20;15]$$

Легко видеть, что члены соответствующие узлам А и С равны ∞. Действительно, узел А не соединён ветвью с узлом С, а направление от узла А к узлу А бессмысленно.

Примем k=1. Тогда получим элемент дистанционной матрицы

$$\delta^1_{AB} = (\gamma_{AB} + d_{BB}) = 10$$

Принимая последовательно k=2 и k=3, получим

$$\delta^2_{AB} = (\gamma_{AE} + d_{EB}) = 15$$
$$\delta^3_{AB} = (\gamma_{AD} + d_{DB}) = 20$$

Аналогично рассматривая выражения и для других элементов δ_{ii} можно получить следующие дистанционные матрицы:

$\Delta^1 =$

	A	B	C	D	E
A	∞	10	∞	5	10
B	10	∞	20	∞	5
C	∞	20	∞	15	15
D	5	∞	15	∞	10
E	10	5	15	10	∞

$\Delta^2 =$

	A	B	C	D	E
A	∞	15	25	20	15
B	15	∞	20	15	20
C	25	20	∞	25	25
D	20	15	25	∞	15
E	15	20	25	15	∞

$\Delta^3 =$

	A	B	C	D	E
A	∞	20	30	25	15
B	40	∞	30	35	35
C	30	30	∞	35	25
D	35	35	25	∞	30
E	15	25	25	20	∞

$\Delta^4 =$

	A	B	C	D	E
A	∞	-	-	-	-
B	-	∞	-	-	-
C	-	-	∞	-	-
D	-	-	-	∞	
E	35	35	30	30	∞

и матрицу маршрутов

$M_A =$

	B	C	D	E	
B_{AB}	1	3	3	2	(3)
B_{AE}	2	2	2	1	
B_{AD}	3	1	1	3	(2)

В связи с тем. что четыре исходящих направления имеет только один $УК_E$, в матрице Δ^4 заполнены элементы только в строке E, на местах остальных элементов указаны прочерки. Цифры в скобках в матрице M_A означают, что могла быть взята другая очерёдность выбора направлений в $УК_A$ при связи с $УК_E$, так как пути, проходящие через узлы B и D, имеют одну и ту же длину.

2. Вопросы для самоподготовки

1. Что понимается под кратчайшим путём?

2. Что такое длина пути? Укажите причину выбора в качестве основного критерия длины пути число транзитных узлов (ветвей) в сети.
3. Чем отличаются между собой матрицы непосредственных связей и матрица длин ветвей?
4. Что такое модернизированная матрица, почему в ней диагональные элементы равны x?
5. Каким образом можно получить план распределения информации в сети?
6. Укажите различия между разовыми и групповыми, детерминированными и статистическими способами выбора плана распределения информации.

3. Задание по работе

1. Ознакомиться с теоретической частью, используя дополнительную литературу, ответить на вопросы самоподготовки.
2. Выбрать вариант задания по указанию преподавателя.
3. По матрице длин ветвей построить граф связи.
4. Найти длины кратчайших путей:
    a. методом рельефов;
    b. матричным методом.
5. Построить дерево путей.
6. Вычислить дистанционные матрицы и матрицы маршрутов.
7. По указанию преподавателя изменить сеть (полагая, что какая-то ветвь вышла из строя) и рассчитать кратчайший путь по неправильному рельефу.
8. Скорректировать рельеф и рассчитать кратчайший путь.

Матрица длин ветвей представляется в общем виде как

L=

	A	B	C	D	E	K
A	I_{AA}	I_{AA}	.	.	.	I_{AK}
B	I_{BA}	I_{BB}	.	.	.	I_{BK}
C	.	.	.	.	.	.
D	.	.	.	.	.	.
E	.	.	.	.	.	.

K	I_{KA}	I_{AB}	.	.	.	I_{KK}

Если в таблице какая-то буква, соответствующая обозначению строки и столбца, равна нулю, то этой строки и столбца нет (будет другая размерность матрицы).

Кроме того, даны только элементы ниже главной диагонали, так как матрицы симметричные, а диагональные элементы равны нулю.

4. Требования к отчёту

1. Отчёт должен содержать:
2. Краткие теоретические сведения.
3. Задание.
4. Результаты выполнения задания вместе с расчётами
5. Обобщающие выводы по результатам расчёта.

Таблица вариантов

Элемент	Номер варианта													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
L_{BA}	10	15	20	10	10	15	5	4	8	30	25	5	5	6
L_{CA}	10	15	15	15	15	15	20	6	6	7	8	25	5	6
L_{CB}	10	5	5	20	-	15	20	6	6	7	8	25	5	6
L_{DA}	10	5	20	25	20	10	20	-	1	7	8	-	5	6
L_{DB}	10	8	15	30	10	10	-	6	5	-	4	30	-	10
L_{DC}	10	15	5	5	30	10	15	6	8	10	4	30	5	-
L_{EA}	10	5	-	1	5	10	15	6	-	10	4	30	5	10
L_{EB}	10	-	20	-	10	20	10	4	6	5	-	10	5	10

L_{EC}	10	10	15	25	20	-	5	-	8	5	5	10	-	20
L_{ED}	10	15	5	15	25	20	8	10	10	-	5	10	5	20
L_{KA}	10	5	-	5	10	-	8	10	20	15	7	15	5	-
L_{KB}	10	15	15	-	8	5	10	25	25	15	7	15	5	15
L_{KC}	10	10	5	15	10	5	-	25	20	15	7	15	5	15
L_{KD}	10	20	20	15	5	5	10	25	15	20	-	6	5	15
L_{KE}	-	30	15	10	-	10	10	10	10	20	10	6		10
Приравнять к нулю	KC	A	B	C	D	E	K	KA	B	C	D	E	K	CD

Литература

5. Лазарев В.Г.. Долганов А.В. Основы теории информационных сетей М.: Связь, 1973. - 264 е.: ил.
6. Лазарев В.Г., Лазарев Ю.К. Динамическое управление потоками ин формации в сетях связи. - М.: Радио и связь, 1983. - 216 е.: ил.
7. Лазарев В.Г. Электронная коммутация и управление в узлах связи. -М.: Связь, 1974.-272 е.: ил.
8. Морозов В.К.. Долганов А.В. Основы теории информационных сетей М.: Высш. шк,, 1987.
9. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.
10. Одом, Уэнделл. Официальное руководство Cisco по подготовке к сертифицированным экзаменам CCENT/CCNA ICND1. 2-е изд. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011 – 672с.
11. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.

12. Четвериков В.И. Подготовка и телеобработка данных в АСУ.М.: Высш. шк,, 1981.- 274 е.: ил.
13. Якубайтис Э.А. Информационно-вычислительные сети. М.: Высш. шк,, 1984. - 325 е.: ил.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

Кафедра информационных систем управления

С. С. Пашин

КОРРЕКТИРУЮЩИЕ КОДЫ

Методические указания к лабораторной работе

по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток

2013

В РАБОТЕ ОТРАЖЕНЫ ОСНОВНЫЕ СВЕДЕНИЯ О КОРРЕКТИРУЮЩИХ КОДАХ, МЕТОДАХ КОДИРОВАНИЯ И ДЕКОДИРОВАНИЯ СООБЩЕНИЙ ПРИ ПЕРЕДАЧЕ ПО КАНАЛАМ СВЯЗИ.

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

ВВЕДЕНИЕ

Цель указаний: ознакомление с общими методами формирования кодов, получение практических навыков по формированию корректирующих кодов на примере построения и декодирования кода Хемминга и систематического кода, изучение свойств этих кодов.

Работа состоит из двух частей: первая посвящена изучению методов построения кодов, вторая - методам декодирования.

В ходе выполнения работы проводится программный контроль знаний студентов.

1. Кодирование информации

Кодирование - процесс преобразования сообщений в комбинации из дискретных сигналов. Код - совокупность правил, в соответствии с которыми производятся данные преобразования.

Каждая комбинация записывается в виде последовательности, составленной из некоторых условных символов - элементов кодовой комбинации. В качестве элементов кодовой комбинации могут использоваться буквы и цифры.

Каждому сообщению однозначно соответствует определённая кодовая комбинация. Код позволяет записать все сообщения на некотором общем для данного набора языке. Набор элементов данного кода рассматривается как алфавит, а кодовые комбинации из этих элементов - как кодовые слова. Каждое сообщение передаётся собственным кодовым словом.

Коды и кодовые комбинации имеют различные характеристики, например структурные:

- число кодовых признаков, используемых для комбинирования;
- количество разрядов кодовой комбинации;
- способ комбинирования (закон, по которому из единичных элементов образуются кодовые комбинации).

По числу кодовых признаков (символов) коды делятся:

- на единичные (используется только 1 символ, кодовые комбинации отличаются друг от друга количеством символов);
- на двоичные (кодовые комбинации строятся из двух символов: 0 и 1);
- многопозиционные (имеют количество символов более двух).

По количеству разрядов кодовые комбинации делятся:

- на равномерные (каждая кодовая комбинация содержит одинаковое, постоянное число разрядов);
- неравномерные (число разрядов кодовых комбинаций различно, непостоянно).

По способу комбинирования различают коды:

- использующие все возможные комбинации;
- частично использующие возможные комбинации.

Кроме различия кодов по перечисленным характеристикам они могут иметь *равное назначение* и в соответствии с этим подразделяются: на телеграфные, телемеханические, телевизионные, коммерческие, военные и т.д.

Наиболее широко в различных автоматизированных устройствах обработки, хранения и передачи информации используются *двоичные коды*. Они делятся на две самостоятельные группы:

- *не избыточные*, или *простые*, или *первичные коды* (используются все возможные комбинации);
- *избыточные* (используют лишь определённую часть возможных кодовых комбинаций, оставшаяся часть используется для обнаружения и исправления ошибок, возникающих при передаче сообщений).

В последних разряды можно разделить на *информационные* - определённое число разрядов, используемых для информации; *проверочные* - разряды, предназначенные для коррекции ошибок.

Обе эти группы делятся на *равномерные* и *неравномерные.* Неравномерные избыточные коды не нашли распространения из-за сложностей в технической реализации.

2. Основные характеристики кодов

Применяются для оценки кодов и выражают количественные и качественные показатели. Они используются при выборе кодов, предназначенных для передачи, хранения и обработки информации.

1. **n**, длина кода - число разрядов (символов), составляющих кодовую комбинацию.
2. **m**, основание кода - количество отличающихся друг от друга значений импульсных признаков, используемых в кодовых комбинациях (для двоичных кодов, где значения импульсных признаков - цифры 0 и 1, m=2).
3. $\mathbf{N_p}$, мощность кода - число кодовых комбинаций (рабочих кодовых слов), используемых для передачи сообщений.
4. $\mathbf{N = m^n}$, полное число кодовых комбинаций - число всех возможных комбинаций.
5. **k**, число информационных символов - количество разрядов кодовой комбинации, предназначенных для передачи собственно сообщения, очевидно,

$$N_p = 2^k \tag{1}$$

6. **r**, число проверочных символов - число символов (разрядов) кодовой комбинации, необходимых для коррекции ошибок.
7. **R**, избыточность кода - относительная избыточность, равная отношению числа проверочных символов к длине кода:

$$R = \frac{r}{n} \tag{2}$$

а в более общем виде:

$$R = 1 - \frac{\log_m N_p}{\log_m N} \tag{3}$$

8. R', скорость передачи – отношение числа информационных разрядов к длине кода:

$$R' = \frac{k}{n} \tag{4}$$

поскольку n=k+r, то R' = 1 – R.

Пример. Определить избыточность и скорость передачи двоичного кода, предназначенного для передачи 16 сообщений, если его длина N = 5.

Полное число кодовых комбинаций: $N = м^N = 2^5 = 32$ =>

$$R = 1 - \frac{\log_m N_p}{\log_m N} = 1 - \frac{\log_2 16}{\log_2 32} = 1 - 4 / 5 = 0{,}2 \Rightarrow$$

$R' = 1 - R = 1 - 0{,}2 = 0{,}8$

9. Ω, вес кодовой комбинации - количество единиц в кодовой комбинации.

Пример. Для кодовой комбинации 101100110 длина кода N=9 и вес Ω=5.

10. **D**, кодовое расстояние между двумя кодовыми комбинациями - число одноимённых разрядов с различными символами; практически кодовое расстояние определяется как вес суммы по модулю два кодовых комбинаций.

Пример. Определить кодовое расстояние между комбинациями 10010111 и 00100110.

Находим: 10010111 ⊕ 00100110 = 10110001. В полученной кодовой комбинации 4 единицы, т.е. со = 4 => D = 4.

11. **W,** весовая характеристика кода - число кодовых комбинаций веса Ω.

Пример. Для кода, содержащего кодовые комбинации 00000, 01110, 10101,11011, весовая характеристика W(0) = 1, W(3) = 2, W(4) = 1, т.е. данный код состоит из одного кодового слова веса 0, двух слов веса 3 и одного слова веса 4.

12. $\mathbf{P_{н.о}}$, вероятность необнаружения ошибки - вероятность такого события, при котором принятая кодовая комбинация отличается от переданной, а свойства данного кода не позволяют определить факт наличия ошибки;

13. **Оптимальность кода** - свойство такого кода, который обеспечивает наименьшую вероятность необнаружения ошибки среди всех кодов той же величины **N** и избыточности **R**;

14. $\mathbf{К_Л(D)}$, коэффициент ложных переходов - коэффициент, показывающий, какая доля кратности **D** не обнаруживается:

$$К_л(\mathrm{d}) = \frac{1}{\mathrm{N_p}} \sum_{\mathrm{i=1}}^{\mathrm{n}} \frac{\mathrm{N_{pi}}}{\mathrm{C_n^d}} \qquad (5)$$

ГДЕ $\mathbf{N_{PI}}$ - ЧИСЛО РАБОЧИХ КОДОВЫХ, КОМБИНАЦИЙ, ОТСТОЯЩИХ ОТ I - КОДОВОЙ КОМБИНАЦИИ НА РАССТОЯНИЕ **D;** C_n^d - ЧИСЛО СОЧЕТАНИЙ ИЗ **N** ПО **D.**

ДЛЯ СИСТЕМАТИЧЕСКИХ КОДОВ ВСЕ КОДОВЫЕ СЛОВА ИМЕЮТ ОДИНАКОВОЕ РАСПРЕДЕЛЕНИЕ КОДОВЫХ РАССТОЯНИЙ ДО ДРУГИХ СЛОВ, ПОЭТОМУ РАСПРЕДЕЛЕНИЕ КОДОВЫХ РАССТОЯНИЙ ДЛЯ ЛЮБОГО СЛОВА МОЖНО ОПРЕДЕЛИТЬ, ИСПОЛЬЗУЯ ВЕСОВУЮ ХАРАКТЕРИСТИКУ:

$$K_л(\omega) = \frac{W(\omega)}{N_n^{\omega}} \qquad (6)$$

ПРИМЕР. ОПРЕДЕЛИТЬ $\mathbf{K_л(D)}$ ДЛЯ КОДА С 4-МЯ РАБОЧИМИ КОМБИНАЦИЯМИ $N_р=4$: $V_1=001$, $V_2=010$, $V_3=011$, $V_4= 100$.

СОСТАВИМ ТАБЛИЦУ КОДОВЫХ РАССТОЯНИЙ МЕЖДУ КАЖДЫМИ КОМБИНАЦИЯМИ:

V	V_1	V_2	V_3	V_4
V_1	0	2	1	2
V_2	2	0	1	2
V_3	1	1	0	3
V_4	2	2	3	0

Из таблицы видно, что Vi удалена на d = 1 от одного вектора и на d = 2 от двух, т.е. $N_{р1}(l)$ -1, $N_{р1}(2) = 2$, $N_{р1}(3) = 0$.

АНАЛОГИЧНО:

$N_{Р2}(L) = 1$, $N_{Р2}(2) = 2$, $N_{Р2}(3) = 0$;

$N_{Р3}(L) = 2$, $N_{Р3}(2) = 0$, $N_{Р3}(3) = L$;

$N_{Р4}(L) = 0$, $N_{Р4}(2) = 2$, $N_{Р4}(3) = L$.

ОПРЕДЕЛИМ КОЭФФИЦИЕНТЫ ЛОЖНЫХ ПЕРЕХОДОВ:

$$K_л(1) = \frac{1}{4} \cdot \frac{N_{р1}(1) + N_{р2}(1) + N_{р3}(1) + N_{р4}(1)}{C_3^1} = \frac{1}{4} \cdot \frac{4}{3} = \frac{1}{3}$$

$$K_л(2) = \frac{1}{4} \cdot \frac{N_{р1}(2) + N_{р2}(2) + N_{р3}(2) + N_{р4}(2)}{C_3^2} = \frac{1}{4} \cdot \frac{6}{3} = \frac{1}{2}$$

$$K_л(3) = \frac{1}{4} \cdot \frac{N_{р1}(3) + N_{р2}(3) + N_{р3}(3) + N_{р4}(3)}{C_3^3} = \frac{1}{4} \cdot \frac{2}{1} = \frac{1}{2}$$

Пример. Пусть весовая характеристика кода длины **n** = 5 следующая:

$W(L) = L$, $W(3) = 2$, $W(4) = 1$, $W(5) = 1$.

Тогда из формулы следует

$$K_л(\omega) = \frac{W(\omega)}{N_n^{\omega}},$$

что $К_л(1) = 0{,}2$; $К_л(2) = 0$; $К_л(3) = 0{,}2$; $К_л(4) = 0{,}2$; $К_л(5) = 1$.

Таким образом, данный код обнаруживает все двукратные ошибки и 80% всех остальных, кроме пятикратных. Пятикратные ошибки не обнаруживаются.

3. Корректирующие коды

Корректирующие коды - коды, позволяющие обнаруживать и исправлять ошибки, возникающие при передаче под воздействием помех. Эти коды имеют некоторую избыточность.

Построение таких кодов осуществляется с помощью единичной матрицы размерностью N

$$E_n = \begin{vmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{vmatrix} \quad (7)$$

Все строки матрицы линейно независимы, так как любая их линейная комбинация не равна нулю. Матрица является определяющей, поскольку сложение по модулю 2 всевозможных сочетаний строк дает $N = 2^N - 1$ ненулевых комбинаций кода.

Ошибки при передаче двоичного кодовой комбинации возникают из-за замены символов на неверные (1 на 0 и наоборот). С целью обнаружения ошибок в коде, имеющем блочную структуру, для передачи информации используют не все 2^N возможных комбинаций, а лишь часть разрешённых комбинаций.

$$N_0 = 2^k, k < n \quad (8)$$

Остальные $2^N - 2^K$ запрещённые комбинации служат для обнаружения ошибок. Появление запрещённой комбинации является следствием искажения разрешённой, т.е. результатом ошибки. Если же искажение одной разрешённой

комбинации приводит к появлению другой разрешённой, то ошибка не обнаруживается.

Механизм искажения двоичной комбинации, т.е. переход к множеству запрещённых комбинаций B_{IT}, можно представить как процесс сложения по модулю 2 векторов кодовой комбинации A_I и ошибок ε_t:

$$B_{it} = A_i \oplus \varepsilon_t , \tag{9}$$

где ε_t - все возможные сочетания ошибок (число единиц в них равно кратности ошибок!).

В n - элементном двоичном простом коде, в котором в качестве разрешённых используются все возможные комбинации двоичных цифр, кодовое расстояние d = 1 и любая из одиночных ошибок переводит одну из разрешённых комбинаций в другую, поэтому ошибка не обнаруживается. Для выявления всех без исключения одиночных ошибок необходимо, чтобы d > 2.

Для обнаружения ошибок кратности $\mathbf{T_0}$:

$$d >= t_0 + 1 \tag{10}$$

Для случая исправления и обнаружения:

$$d >= t_0 + t_И + 1 \tag{11}$$

Исправляемые ошибки кратности $\mathbf{T_И}$ связаны с кодируемым расстоянием:

$$d = 2 \cdot t_И + 1 \tag{12}$$

Для ориентировочного округления необходимой избыточности кода при заданном d пользуются оценкой Хемминга:

$$n - k \geq \log_2 \sum_{t=0}^{t_n} C_n^t \tag{13}$$

или для практических расчётов:

$$n - k \geq \log_2 \frac{n^t + n^{t-1} + K + 1}{t!} .$$

Сообщения передаются в виде сигналов, имеющих определенную форму и последовательность. В телеграфе сообщение обычно передается при помощи алфавитов, цифр или алфавита и цифр вместе. Сигналы следуют в определенной последовательности. Например, в коде Морзе каждой букве и цифре соответствует

некоторая последовательность коротких (точки) и длинных (тире) посылок тока, разделяемых кратковременными паузами по длительности такими же, как и точки. Пробел между буквами при этом изображается выключением тока на три единицы времени, а пробел между словами - на шесть единиц времени (рис. 1).

Если обозначить тире 1, а точки 0, то можно записать:

А	Б	В	Г	Д
01	1000	011	110	100

По коду Бодо, применяемому в современных буквопечатающих аппаратах, каждой букве соответствует сигнал из пяти импульсов одинаковой длительности и формы (рис. 2).

А	Б	В	Г	Д
1	0	0	01	11

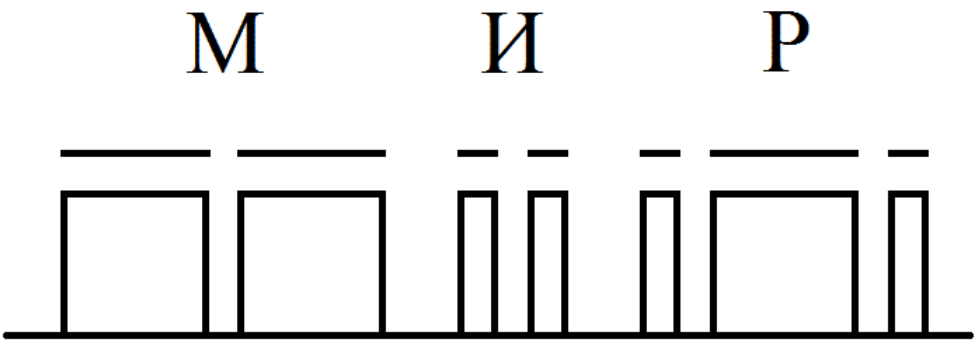


Рис. 1 Код Морзе

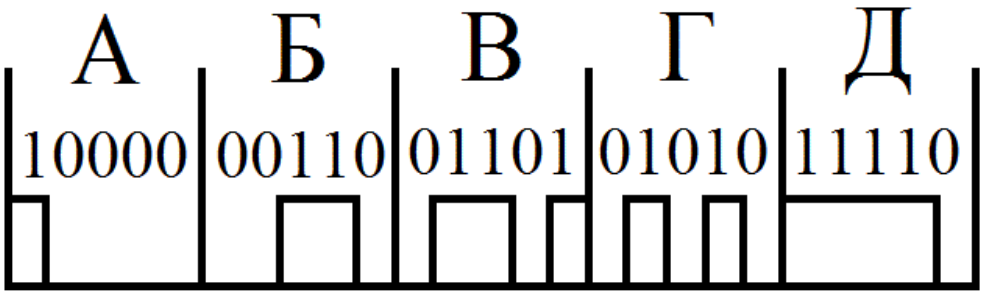


Рис. 2 Код Бодо

Следует различать *способ кодирования* и *способ модуляции* сигнала или сообщения. Так, рассмотренные коды Морзе и Бодо двоичные, т. е. имеют двойное

основание: сообщения передаются с помощью посылки сигнала (тока) или его отсутствия. Могут быть и другие коды.

Вообще *кодированием* называется *отображение состояния одной физической системы с помощью состояния другой.*

Цель кодирования состоит в том, чтобы предоставить информацию в более компактной и удобной форме для оперирования при передаче и обработке информации; приспособить кодированную информацию к обработке на конкретных электронных устройствах. Декодирование является операцией, обратной кодированию.

Совокупность условных символов или сигналов, обозначающих определенное сообщение, называется *кодом*. Код обычно представляется в виде таблицы, в которой каждому сообщению поставлено в соответствие кодовое обозначение.

Код строится из элементов. Число различных элементов называется *основанием кода*. Код с основанием два называется *двоичным*, код с основанием три - *троичным* и так далее. Наиболее широкое практическое применение получили двоичные коды с элементарными символами 0 и 1.

Условные сигналы, составляющие код, называются кодовыми *комбинациями (кодовыми словами)*. Число элементов или знаков, образующих кодовую комбинацию, называются *значностью кода*.

Коды, кодовые комбинации которых имеют различное число знаков, называются *неравномерными*. Неравномерный код должен быть непременно обратимым (префиксным). Кодирование будет обратимым всякий раз, когда выполняются следующие два условия:

а) кодовые комбинации различны;

б) любая кодовая комбинация не является началом другой.

Если не выполняется хотя бы одно условие, код называется *необратимым*. Необратимый код требует специальных разделительных, знаков которые ставятся между кодовыми комбинациями для того, чтобы код можно было декодировать (как в коде Морзе).

Двоичный код характеризуется наличием двух различных элементов, правда физический характер этого различия не играет никакой роли. Строение кода удобно представлять в виде кодового дерева. Из каждой вершины выходит число ребер

(ветвей), равное основанию кода. Если код двоичный, то с каждой вершины кодового дерева выходит два ребра. Условимся шаг влево обозначать символом нуль, шаг вправо - символом единица.

В качестве примера рассмотрим строение трех кодов: равномерного (А), неравномерного обратимого (В) и неравномерно - необратимого (С). Для построения кода использовано восемь сообщений от а до н.

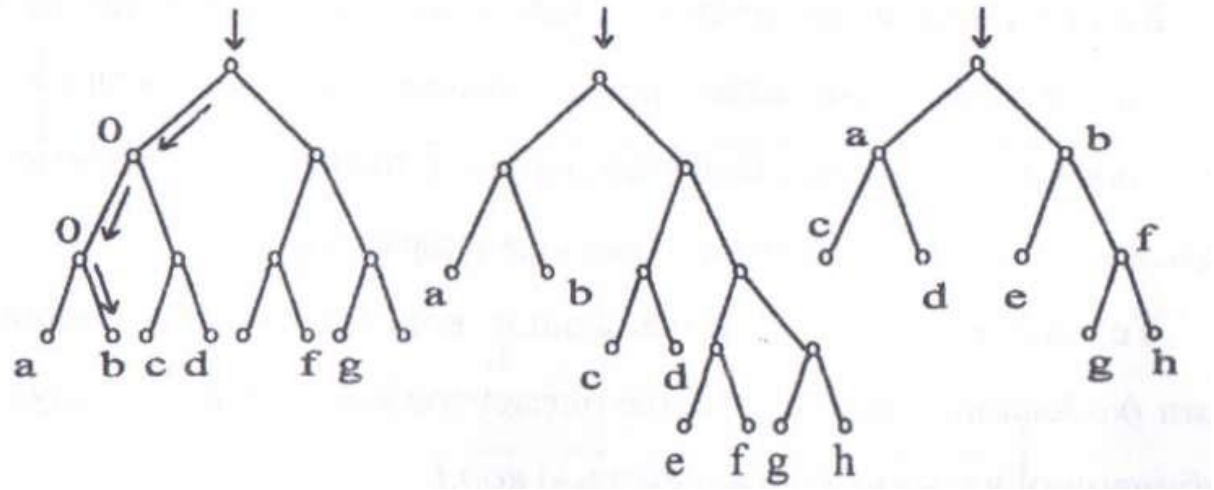


Рис. 3. Коды: А - неравномерный; В - неравномерный обратимый;
С - неравномерный необратимый.

Кодовые обозначения сведены в таблицу.

Кодовые обозначения Таблица 1

	A	B	C	D	E	F	G	H
A	000	001	010	011	100	101	110	111
B	00	01	100	101	1100	1101	1110	1111
C	0	1	00	01	10	11	100	111

Закодированные последовательности сообщений легко декодируются при помощи кодового дерева. Закодируем последовательность сообщений при помощи кода А. Полученная последовательность символов 001010011 легко расшифровывается, если учесть, что шаг влево означает нуль, а шаг вправо единицу. На кодовом дереве «код А» показано прохождение кодовой комбинации

001. Одно и то же сообщение можно закодировать разными способами. Возникает вопрос об оптимальных (на выгоднейших) способах кодирования.

Естественно считать оптимальным такой код, при котором на передачу сообщения заданной длины будет затрачено минимальное количество элементарных символов.

4. Систематические коды

Систематические коды - групповой n - значный код, в котором из n символов, образующих кодовую комбинацию, к символов - информационные, а r = n-k - избыточные, предназначенные для проверки. Эти коды задаются с помощью производящей и проверочной матриц.

Производящая матрица позволяет получить все возможные комбинации кода суммированием по модулю 2 всех возможных сочетаний строк. Её образуют приписыванием к единичной матрице размерности к дополнительной матрицы размерности k x r.

$$A = \left[\begin{array}{\overbrace{ccccc}^{k} \mid \overbrace{ccccc}^{r = n - k}} 1 & 0 & \dots & 0 & 0 & a_{11} & a_{12} & \dots & a_{1r-1} & a_{1r} \\ 0 & 1 & \dots & 0 & 0 & a_{21} & a_{22} & \dots & a_{2r-1} & a_{2r} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 & a_{k-11} & a_{k-12} & \dots & a_{k-1r-1} & a_{k-1r} \\ 0 & 0 & \dots & 0 & 1 & a_{k1} & a_{k2} & \dots & a_{kr-1} & a_{kr} \end{array}\right]$$

Строки дополнительной матрицы получают перебором различных разрядных комбинаций, содержащих не менее d - 1 единиц, причём сумма по модулю 2 двух любых строк должна содержать не менее d - 2 единиц.

$$H = \left[\begin{array}{cccccccccc} \multicolumn{5}{c}{\text{k}} & \multicolumn{5}{c}{\text{r = n - k}} \\ a_{11} & a_{12} & \dots & a_{1r-1} & a_{1r} & 1 & 0 & \dots & 0 & 0 \\ a_{21} & a_{22} & \dots & a_{2r-1} & a_{2r} & 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{k-11} & a_{k-12} & \dots & a_{k-1r-1} & a_{k-1r} & 0 & 0 & \dots & 1 & 0 \\ a_{k1} & a_{k2} & \dots & a_{kr-1} & a_{kr} & 0 & 0 & \dots & 0 & 1 \end{array}\right]$$

Проверочная матрица получается приписыванием к единичной матрице размерности r = n - k транспонированной дополнительной матрицы r x k (т.е. строки формируются из столбцов дополнительной матрицы).

Затем для строк составляются уравнения проверки - в строке сумма по модулю 2 A_I разных единице должна быть равна 0. Проверочные разряды записаны в уравнениях последними.

Искажение одного из элементов нарушит эти уравнения, по которым можно определить и восстановить искажённый элемент. Такая проверка является проверкой на чётность по определённому сочетанию разрядов.

Коды проверочных разрядов получаем из уравнений проверки.

Пример: Построить систематический код длины n = 7 (если задано количество информационных разрядов k, то n определяется из оценки Хемминга), исправляющий ошибки кратности $\tau_И = 1$.

Определяем кодовое расстояние $d = 2\tau_И + 1 = 3$. Число проверочных символов из оценки Хемминга :n-k = 3 = r, k = 4.

Производящая матрица:

$$A = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]$$

Проверочная матрица:

$$H = \left[\begin{array}{cccc|ccc} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Уравнения проверки:

Коды цифр проверочных разрядов:

$$r_1 = a_2 \oplus a_3 \oplus a_4 \oplus a_5 = 0 \qquad a_5 = a_2 \oplus a_3 \oplus a_4$$
$$r_2 = a_1 \oplus a_3 \oplus a_4 \oplus a_6 = 0 \qquad a_6 = a_1 \oplus a_3 \oplus a_4$$
$$r_3 = a_1 \oplus a_2 \oplus a_4 \oplus a_7 = 0 \qquad a_7 = a_1 \oplus a_2 \oplus a_4$$

Данный алгоритм декодирования трудно применим при программной реализации декодирующего устройства, поэтому при моделировании на ЭВМ декодирующего устройства предпочтительно применять принцип декодирования по максимуму правдоподобия. В этом случае проверочная матрица H строится приписыванием снизу к правой части производящей матрицы A единичной матрицы E.

При прохождении кодовой последовательности A по каналам связи она может искажаться за счёт наложения шумовой последовательности Z_i , т.е. кодовая последовательность Y на выходе канала связи определяется как

$$Y_i = A_i \oplus Z_i \qquad (14)$$

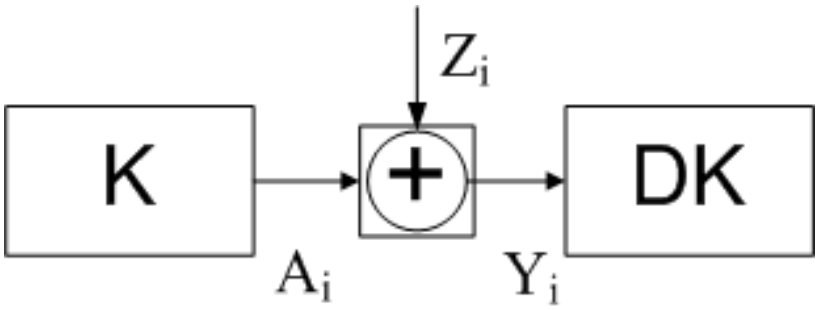


Рис. 4 Кодовая последовательность

Задача восстановления переданной информационной последовательности сводится к нахождению шумовой последовательности Z_I такой, что $A_i \oplus Z_i = Y_i$, тогда $A_i = Y_i \oplus Z_i$.

Однако для любой принятой последовательности Y_I существуют различные пары A_I и Z_I, удовлетворяющие уравнению (14). Задача декодирования (в соответствии с принципом декодирования по максимуму правдоподобия) сводится к нахождению шумовой последовательности Z_I с минимальным весом.

Для решения этой задачи наиболее просто воспользоваться таблицей декодирования, основанной на том, что:

$$Y_i \times H = (A_i \oplus Z_i) \times H = A_i \times H \oplus Z_i \times H = Z_i \times H = S_i \quad (15)$$

где S_I - синдром.

Количество возможных синдромов 2^m , где m - количество проверочных символов. Таблица декодирования W ставит в соответствие каждому синдрому шумовую последовательность. Для того чтобы построить таблицу декодирования, перебирают все шумовые последовательности, начиная с наименьшего Z_I веса.

Процесс декодирования заключается в определении по формуле (15) синдрома, выборе по синдрому соответствующей шумовой последовательности из таблицы W и восстановлении информационной последовательности.

Пример: рассмотрим процесс декодирования на примере, разобранном выше: n = 7, $t_И$ = 1.

Проверочная матрица

$$H = \begin{vmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Декодирующая матрица

$$W = \begin{vmatrix} \multicolumn{7}{c|}{Z} & \multicolumn{3}{c}{S} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{vmatrix}$$

Процесс декодирования: Пусть A_i = 1000011, а Y_i = 1100011

1. Находим синдром $S_i = Y_i$ x H = 101
2. По таблице декодирования находим синдром Z_i = 0100000
3. Восстанавливаем информационную последовательность $A_i = Y_i \oplus Z_i$ =1000011

5. Код Хемминга

Код Хемминга представляет собой один из наиболее распространённых систематических кодов, имеющих простой и удобный для технической реализации алгоритм обнаружения и исправления одиночной ошибки.

При построении уравнений проверки для кода Хемминга все номера

разрядов кода записывают в двоичном виде: $a_1=a_{0001}$, $a_2=a_{0010}$, $a_3=a_{0011}$ и т.д.

Код Хемминга строится так, чтобы полученный при проверках результат $(r_1,r_2,...,r_{n-k})$ прямо указал номер искажённого разряда и тем самым упростил декодирование. Для этого необходимо, чтобы при первой проверке получилась цифра с весом младшего разряда числа 2^0, при второй проверке - с весом следующего разряда 2^1 и т.д. Таким образом, проверочные уравнения составляют как сумму по модулю 2 всех разрядов, в номерах которых в соответствующем разряде 2^0 , 2^1 , 2^2 и т.д. стоит единица:

$$r_1 = a_1 \oplus a_2 \oplus a_3 \oplus a_4 \oplus a_5 \oplus \dots$$
$$r_2 = a_2 \oplus a_3 \oplus a_4 \oplus a_5 \oplus a_6 \oplus \dots$$
$$r_3 = a_3 \oplus a_4 \oplus a_5 \oplus a_6 \oplus a_7 \oplus \dots$$

УРАВНЕНИЯ КОДИРОВАНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ ПРОВЕРОЧНЫХ РАЗРЯДОВ НАХОДЯТ ПРИРАВНИВАНИЕМ ПРОВЕРОЧНЫХ УРАВНЕНИЙ К НУЛЮ ПРИ ОТСУТСТВИИ ОШИБОК.

ПРОВЕРОЧНЫЕ РАЗРЯДЫ РАЗЛИЧАЮТСЯ ВНУТРИ КОДОВОЙ КОМБИНАЦИИ НА МЕСТАХ, СООТВЕТСТВУЮЩИХ ИХ НОМЕРУ/

Пример: Составить код Хемминга для к = 6:

Двоичные номера	Уравнения проверки	Проверочные разряды
a_{0001}	$r_1 = a_1 \oplus a_3 \oplus a_5$	$b_1 = a_3 \oplus a_5$
a_{0010}	$r_2 = a_2 \oplus a_3 \oplus a_6$	$b_2 = a_3 \oplus a_6$
a_{0011}	$r_3 = a_4 \oplus a_5 \oplus a_6$	$b4 = a_5 \oplus a_6$
a_{0100}		

a_{0101}		
a_{0110}		

Маска кода

1	2	3	4	5	6	7	8	9
b_1	b_2	a_1	b_4	a_2	a_3	a_4	a_5	a_6

6. Код Шеннона - Фано

С учетом статических характеристик поступающих сообщений метод построения эффективного кода был предложен давно. В связи с тем, что его методика существенно не отличается от методики построения эффективного кода, предложенного Шенноном, код получил название кода Шеннона - Фано.

Согласно методике Шеннона - Фано, построение оптимального кода ансамбля из сообщений сводится к следующему:

1-й шаг. Множество из сообщений располагается в порядке убывания вероятностей.

2-й шаг. Первоначальный ансамбль кодируемых сигналов разбивается на две группы таким образом, чтобы суммарные вероятности сообщений обеих групп были по возможности равны. Если равной вероятности в подгруппах нельзя достичь, то их делят так, чтобы в верхней части (верхняя подгруппа) оставались символы, суммарная вероятность которых меньше суммарной вероятности символов в нижней части (нижняя подгруппа).

3-й шаг. Первой группе присваивается символ 0, а второй группе символ 1.

4-й шаг. Каждую из образованных групп делят на две части таким образом, чтобы суммарные вероятности вновь образованных подгрупп были по возможности равны.

5-й шаг. Первым группам каждой из подгрупп вновь присваивается 0, а вторым 1. Таким образом мы получаем вторые цифры кода. Затем каждая из четырех групп вновь

делится на равные (с точки зрения суммарной вероятности) части до тех пор, пока в каждой из подгрупп не останется по одной букве.

Построение кода Шеннона-Фано рассмотрим на примере кодирования букв русского алфавита. Буквы, расположенные в порядке убывания вероятностей, приведены в табл. 2.

Вероятности расположения букв Таблица 2.

Буква	**Частота**	**Буква**	**Частота**	**Буква**	**Частота**
пробел	0,175	к	0,028	ч	0,012
о	0,09	м	0,026	й	0,01
е, ё	0,072	д	0,025	х	0,009
а	0,062	п	0,023	ж	0,007
и	0,062	у	0,021	ю	0,006
т	0,053	я	0,018	ш	0,006
н	0,053	ы	0,016	ц	0,004
с	0,045	з	0,016	щ	0,003
р	0,04	ъ, ь	0,014	э	0,003
в	0,038	б	0,014	ф	0,002
л	0,035	г	0,013		

Построим код по методике (прилож. 1). Получим код Шеннона-Фано. Этот код обратимый, поэтому декодирование не будет осуществляться однозначно.

Например, фраза "Теория информации" в предложенном коде будет выглядеть так: 1000, 0110, 010, 10110, 01111, 111000, 00, 01111, 1001, 1111111111, 010, 10110, 11001, 01110, 11111110, 01111, 01111. В соответствии с приведенной методикой существует еще определение оптимального кода.

Оптимальным считается код, имеющий минимальную среднюю длину:

$$\lambda_{nд} = \min \tag{16}$$

причем

$$\lambda_{nд} = \sum P_e \lambda_e \tag{17}$$

где суммирование выполняется по всем символам алфавита;

λ_e - длина кодовой комбинации, равная числу ее элементов, соответствующая каждому символу алфавита;

P_e - вероятность появления в сообщениях данного ансамбля каждого символа; $\sum P_e = 1$.

Построенные по этим методикам коды с неравномерным распределением символов, имеющие минимальную среднюю длину кодового слова, называют оптимальными неравномерными кодами (ОНК). Равномерные коды могут быть оптимальными только для передачи сообщений с равновероятным распределением символов первичного алфавита.

Если энтропия источника сообщений не равна максимальной энтропии для алфавита с данным количеством качественных признаков (имеются в виду качественные признаки алфавита, при помощи которых составляются сообщения), то это прежде всего означает, что сообщения данного источника могли бы нести большее количество информации. Абсолютная недогруженность на символ сообщений такого источника:

$$\Delta D = (H_{max} - H) \text{ бит/символ} \tag{18}$$

Для определения количества "лишней" информации, которая заложена в структуре алфавита либо в природе кода, вводится понятие избыточности. Избыточность, с которой мы имеем дело в теории информации, не зависит от содержания сообщения и обычно заранее известна из статических данных. Информационная избыточность показывает относительную недогруженность на символ алфавита и является величиной безразмерной:

$$D = \frac{H_{\max} - H}{H_{\max}} = 1 - \frac{H}{H_{\max}}, \tag{19}$$

где $\frac{H}{H_{\max}} = \mu$ - коэффициент сжатия (относительная энтропия) H и $H_{\max}$, берутся относительно одного и того же алфавита.

Для передачи сообщения достаточно иметь длину кодовой комбинации:

$$L \geq \frac{Log_2 N}{Log_2 m}, \tag{20}$$

где N - общее количество передаваемых сообщений.

L можно представить и как:

$$L \geq \frac{Log_2 m_1}{Log_2 m_2}, \tag{21}$$

где m_1 и m_2 - соответственно качественные признаки первичного и вторичного алфавитов.

Первичный алфавит составлен из m_1 символов (качественных признаков), при помощи которых записано передаваемое сообщение. Вторичный алфавит состоит из m_2, при помощи которых сообщение трансформируется в код, поэтому для цифры 5 в двоичном коде можно записать:

$$L \geq \frac{Log_2 5}{Log_2 2} = 2{,}32$$

Можно ввести понятие *максимально эффективного ОНК*, это такие коды, у которых

$$Log_2 m \sum_{i=1}^{N} P_i \lambda_i = \lambda_{cp} = H \tag{22}$$

Для двоичных кодов:

$$\sum_{i=1}^{N} P_i \lambda_i = \sum_{i=1}^{N} P_i Log_2 P_i \qquad (23)$$

Эффективность ОНК оценивают при помощи *коэффициента статического сжатия*:

$$K_{c.c.} = \frac{H_{max}}{\lambda_{cp}} = \frac{Log_2 N}{Log_2 m \sum_{i=1}^{N} P_i \lambda_i}, \qquad (24)$$

который характеризует уменьшение количества двоичных знаков на символ сообщения при применении ОНК по сравнению с применением методов нестатического кодирования и *коэффициента относительной эффективности*:

$$K_{o.э.} = \frac{H}{\lambda_{cp}}, \qquad (25)$$

который показывает, насколько используется статическая избыточность передаваемого сообщения.

Для примера построим ОНК для передачи сообщений, алфавит которых состоит из двух букв А и Б с вероятностями $P_а$=0,89, $P_в$=0,11 при кодировании по одному, по два и по три символа в блоке. Оценим эффективность полученных кодов.

Оценку эффективности блочного кодирования произведем по табл. 3, предварительно разбив ее по горизонтали на три части по количеству символов в блоке. Рассчитаем произведения вероятностей появления символов на длину блока

Вероятность расположения блоков Таблица 3

Случай кодирования	**Блок**		**1**	**2**	**3**	**4**	**5**	**Число знаков в слове**	
1	А	0,89	0					1	0,89
1	В	0,11	1					1	0,11

2	AA	0,792	0					1	0,792
2	AB	0,098	1	0				2	0,196
2	BA	0,098	1	1	0			3	0,294
2	BB	0,012	1	1	1			3	0,036
3	AAA	0,705						1	0,705
3	AAB	0,087	1	0	0			ч	0,261
3	ABA	0,087	1	0	1			1	0,261
3	BAA	0,087	1	1	0			3	0,261
3	ABB	0,011	1	1	1	0	0	5	0,055
3	BAB	0,011	1	1	1	0	0	5	0,055
3	BBA	0,011	1	1	1	1	1	5	0,055
3	BBB	0,001	1	1	1	1	1	5	0,005

$$L_1 = \sum_i P_i \lambda_i = 0{,}89 + 0{,}11 = 1$$

$$H_1 = -\sum_{i=1} P_i Log_2 P_i = -(0{,}89 \cdot Log_2 0{,}89 + 0{,}11 \cdot Log_2 0{,}11) =$$

$$0{,}3503 + 0{,}1496 = 0{,}4999 \textit{бит / символ}$$

$$L_{11} = \sum_i P_i \lambda_i = 0{,}792 + 0{,}196 + 0{,}294 + 0{,}036 = 1{,}318$$

$$H_{11} = -\sum_{i=1} P_i Log_2 P_i = -(0{,}792 \cdot Log_2 0{,}792 + 0{,}098 \cdot Log_2 0{,}098$$
$$+ 0{,}098 \cdot Log_2 0{,}098 + 0{,}012 \cdot Log_2 0{,}012) = 0{,}9948 \textit{б / с}$$

$$L_{111} = \sum_i P_i \lambda_i = 0{,}705 + 3 \times 0{,}261 + 3 \times 0{,}055 + 0{,}005 = 1{,}658$$

$$H_{111} = -\sum_{i=1} P_i Log_2 P_i = -(0{,}705 \cdot Log_2 0{,}705 + 3 \times 0{,}087 \cdot Log_2 0{,}087 + 3 \times 0{,}011 \cdot Log_2 0{,}011 + 0{,}001 \cdot Log_2 0{,}001) = 1{,}4998 б/с$$

Сравниваем эффективность ОНК в первом и третьем случаях;

$$K_{с.с.1} = \frac{H_{max1}}{\lambda_{ср1}} = \frac{Log_2 2}{1} = 1 \quad K_{с.с.111} = \frac{H_{max111}}{\lambda_{ср111}} = \frac{Log_2 8}{1{,}658} \cong 1{,}809$$

$$K_{о.э.1} = \frac{H_1}{\lambda_{ср1}} = \frac{0{,}499}{1} = 0{,}499 \quad K_{о.э.111} = \frac{H_{111}}{\lambda_{ср111}} = \frac{1{,}4998}{1{,}658} \cong 0{,}932$$

Отметим, что с увеличением числа символов в блоке эффективность кодирования быстро растет (растут $К_{с.с.}$ и $К_{о.э.}$) до определенного предела. Затем рост эффективности постепенно уменьшается. Практика показывает, что с увеличением символов в блоке (n > 4) сложность кодирующих устройств растет быстрее чем эффективность.

Кроме приведенных характеристик можно привести расчет скорости передачи информации. Подсчитаем скорость передачи информации, которая обеспечивается полученным кодом.

Пусть длительность символов кодовых комбинаций равна τ. Тогда средняя длительность кодовых комбинаций:

$$\begin{aligned} T = {} & p(x_1) \times \tau + p(x_2) \times \tau + p(x_3) \times \tau + p(x_4) \times 2\tau + p(x_5) \times 3\tau + \\ & p(x_6) \times 3\tau + p(x_7) \times 3\tau + p(x_8) \times 3\tau + p(x_9) \times 3\tau + p(x_{10}) \times 3\tau + \\ & p(x_{11}) \times 5\tau + p(x_{12}) \times 5\tau + p(x_{13}) \times 5\tau + p(x_{14}) \times 5\tau = 3{,}976\tau \end{aligned} \qquad (26)$$

Средняя энтропия на символ сообщения:

$$H(x) = -\sum_{i=1}^{N} P(x_1) \log_2 p(x_1) = 3{,}976 \frac{дв.ед.}{сообщ} \qquad (27)$$

Таким образом, скорость передачи информации:

$$Y(x) = \frac{1}{\tau} = c \frac{дв.ед.}{с}; \qquad Y(x) = 0{,}3 \frac{дв.ед.}{с} \qquad (28)$$

Следовательно, полученный код в рассматриваемом случае позволил получить максимально возможное значение скорости информации, т.е. обеспечить полное согласование статистических характеристик источника сообщений со свойствами канала.

Это удается тогда, когда значения вероятностей выбираются $P(x_i)$ такими, что условия деления на подгруппы удается выполнить точно. В реальных условиях это, как правило, не обеспечивается и скорость передачи информации будет меньше пропускной способности канала.

Эффективность кодирования может быть при необходимости увеличена путем перехода от кодирования одиночных символов сообщения к кодированию групп символов сообщения, причем с укрупнением групп она повышается.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ 1

1. Ознакомьтесь с теоретической частью, используя дополнительную литературу.
2. На основании полученных у преподавателя исходных данных (количества передаваемых сообщений N), рассчитайте необходимое число информационных и контрольных разрядов для систематического кода, обнаруживающего и исправляющего одиночные ошибки.
3. Составьте систему уравнений кодирования для определения проверочных разрядов для кода Хемминга по п.2.
4. Произведите программный контроль выполнения п.2 и п.3 на примере некоторых случайных сообщений рассчитанной ранее разрядности.
5. Составьте порождающую и проверочную матрицы, а также уравнения проверки по п.2, исходя из заданной преподавателем кратности ошибок $t_И$ и количества информационных разрядов k.

Варианты задания

Вариант	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**
$T_И$	2	3	2	3	1	2	3	2	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3
k	4	4	5	5	6	6	6	3	8	8	8	9	9	9	12	10	10	14	11	11	16	12	7

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ 2

1. Постройте код Хемминга по заданным исходным данным (числу информационных разрядов k) и $t_И = 1$.
2. Постройте статистику ошибок при передаче заданного количества кодовых комбинаций N при заданной вероятности ошибочной передачи кодовой комбинации P
3. По заданной порождающей матрице систематического кода постройте проверочную матрицу по методу наибольшего правдоподобия, уравнения проверки и таблицу декодирования.
4. Постройте статистику ошибок по п.2.

Таблица вариантов задания

Вариант	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
N	200	300	400	500	150	250	350	450	550	200
P	0,02	0,02	0,01	0,02	0,03	0,01	0,02	0,03	0,01	0,02
Вариант	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
N	300	400	500	250	350	450	550	150	300	400
P	0,03	0,05	0,02	0,03	0,04	0,05	0,04	0,05	0,02	0,06

ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ОТЧЕТА

Отчёт должен содержать:

1. Краткие теоретические сведения.
2. Задание.
3. Результаты выполнения задания.
4. Результаты выполнения программного диалога.

5. Выводы о корректирующей способности кода.

ВОПРОСЫ ДЛЯ САМОПОДГОТОВКИ

1. Приведите классификацию корректирующих кодов по способу введения и использования избыточности, по структуре кода. Какие коды среди систематических имеют наибольшую практическую значимость и почему?
2. Какой код называется «оптимальным», в чём заключается сущность оптимального кодирования и практический результат его применения?
3. Какими графическими и геометрическими способами можно представить коды? Приведите пример.
4. Как получают дополнительную матрицу при формировании систематических кодов и чем объясняется такое требование её построения?
5. Какими выражениями удобно пользоваться в практических расчётах для определения числа контрольных разрядов кодов с d = 3? d = 4?
6. В чём состоит недостаток кодов с проверкой на чётность?
7. Что такое кодовое расстояние, как оно определяется между двумя комбинациями двоичного кода? Какова связь корректирующей способности кода с новым расстоянием?

Литература

1. Информационная безопасность телекоммуникационных систем (технические аспекты): учеб. пособие / В.Г. Кулаков, М.В. Гаранин, А.В. Заряев. - М.: Радио и связь, 2004.
2. Кузьмин И.В., Кедрус В.А. Основы теории информации и кодирования: учеб. пособие. - Киев: Высш. шк., 1986.
3. Новиков А.А. Уязвимости и информационная безопастность телекоммуникационных технологий: учеб. пособие. - М. : Радио и связь, 2003. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.

4. Одом, Уэнделл. Официальное руководство Cisco по подготовке к сертифицированным экзаменам CCENT/CCNA ICND1. 2-е изд. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011 – 672с.
5. Сырецкий Г.А. Информатика. Фундаментальный курс. Том 1 - СПб.: БХВ - Петербург, 2005.
6. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.
7. Симонов Б.Б., Плохое Е.М. Теория информации и кодирование: учеб. пособие. - Ростов н/Д. Феникс, 2002.
8. Четвериков В.И. Подготовка и телеобработка данных в АСУ. - М.: Высш. шк.,1981.
9. Шилейко А.В. и др. Введение в информационную теорию систем. - М.: Радио и связь, 1985.

Приложение 1

Кодовая таблица

Буква	Частота	Двоичные разряды									
		1-й	2-й	3-й	4-й	5-й	6-й	7-й	8-й	9-й	10-й
пробел	0,175	0	0								
о	0,09	0	1	0							
е, ё	0,072	0	1	1	0						
а	0,062	0	1	1	1	0					
и	0,062	0	1	1	1	1					
т	0,053	1	0	0	0						
н	0,053	1	0	0	1						

с	0,045	1	0	1	0						
р	0,04	1	0	1	1	0					
в	0,038	1	0	1	1	1	0				
л	0,035	1	0	1	1	1	1				
к	0,028	1	1	0	0	0					
м	0,026	1	1	0	0	1					
д	0,025	1	1	0	1	0					
п	0,023	1	1	0	1	1	0				
у	0,021	1	1	0	1	1	1				
я	0,018	1	1	1	0	0	0				
ы	0,016	1	1	1	0	0	1				
з	0,016	1	1	1	0	1	0				
ъ, ь	0,014	1	1	1	0	1	1	0			
б	0,014	1	1	1	0	1	1	1			
г	0,013	1	1	1	1	0	0				
ч	0,012	1	1	1	1	0	1	0			
й	0,01	1	1	1	1	0	1	1			
х	0,009	1	1	1	1	1	0	0			
ж	0,007	1	1	1	1	1	0	1			
ю	0,006	1	1	1	1	1	1	0	0		
ш	0,006	1	1	1	1	1	1	0	1		
ц	0,004	1	1	1	1	1	1	1	0		
щ	0,003	1	1	1	1	1	1	1	1	0	
э	0,003	1	1	1	1	1	1	1	1	1	0
ф	0,002	1	1	1	1	1	1	1	1	1	1

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК

Кафедра информационных систем управления

С. С. Пашин

МЕЖСЕТЕВЫЕ ЭКРАНЫ. PFSENCE

Методические указания к лабораторной работе

по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток

2013

Приведены основные сведения о фильтрации сообщений в сетях передачи данных, рассмотрена работа межсетевых экранов на примере pfSence. Цель указаний - получение практических навыков развертывания межсетевых экранов, создания правил фильтрации входящих и исходящих соединений.

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

Межсетевые экраны

Основной задачей сетевого экрана является защита компьютерных сетей или отдельных узлов от несанкционированного доступа. Также сетевые экраны часто называют фильтрами, так как их основная задача — не пропускать (фильтровать) пакеты, не подходящие под критерии, определённые в конфигурации.

Некоторые сетевые экраны также позволяют осуществлять трансляцию адресов — динамическую замену внутрисетевых (серых) адресов или портов на внешние, используемые за пределами ЛВС.

pfSence

Проект pfSense возник в 2004 году как форк популярного дистрибутива m0n0wall авторами Chris Buechler и Scott Ullrich. С самого начала он был нацелен на полную установку на компьютер, в противовес нацеленности m0n0wall на встраиваемые системы. Тем не менее, pfSense доступен так же в виде образа для встраиваемых систем на основе CompactFlash. Версия 1.0 была выпущена 4 октября 2006.

На данный момент идет разработка версии 2.1, которая принесет массу улучшений. Включая инструменты для централизованного управления большим количеством pfSense систем. Версия 2.1 содержит полную поддержку IPv6 в основных сервисах.

pfSense это дистрибутив для создания межсетевого экрана/маршрутизатора, основанный на FreeBSD. pfSense предназначен для установки на персональный компьютер, известен своей надежностью и предлагает функции, которые часто можно найти только в дорогих коммерческих межсетевых экранах. Настройки можно проводить через web-интерфейс что позволяет использовать его без знаний базовой системы FreeBSD. pfSense обычно применяется в качестве периметрового брандмауэра, маршрутизатора, сервера DHCP/DNS, и в качестве VPN hub/spoke.

Название происходит от факта, что pfSense помогает использовать инструмент фильтрации пакетов pf из OpenBSD более осознано для непрофессиональных пользователей

Основные возможности

- Firewall
- State Table
- NAT — Network Address Translation
- Redundancy — два или более файрвола могут быть объединены в отказоустойчивую группу, также поддерживается синхронизация настроек между ними
    - CARP — CARP из OpenBSD позволяет создать аппаратную защиту от сбоев. Два или более межсетевых экрана могут быть объединены в отказоустойчивую группу. В случае отказа сетевого интерфейса на главном межсетевом экране, активным становится другой. Так же pfSense предоставляет возможность синхронизации настроек: если изменены настройки на одном фаерволле, то они автоматически будут синхронизированы на другом.
    - pfsync — pfsync обеспечивает репликацию состояния фаерволлов. Это означает, что все существующие сетевые соединения сохранятся при выходе из строя одного из фаерволлов, что очень важно для обеспечения отказоустойчивости сети.
- Outbound and Inbound Load Balancing
- VPN сервер — IPsec, OpenVPN, PPTP
- PPPoE сервер
- Динамический DNS
- DHCP сервер и шлюз
- Прокси сервер
- Captive portal — перенаправление на специальную веб-страницу для авторизации для доступа в Интернет
- Мониторинг и графические отчеты с использованием RRD
- Работа в режиме LiveCD
- Поддержка программных модулей.

Наиболее значимые расширения:

- Squid — прокси-сервер

- Snort — система обнаружения/нейтрализации вторжений.

Брандмауэр

Основная и главная функция pfSense в любой разворачиваемой системе - фильтрация трафика. В данной части руководства мы рассотрим основные принципы работы брандмауера, а также покажем наиболее правильные методы конфигурирования правил.

Основные принципы работы брандмауэра

Далее мы познакомим Вас с основными терминами и понятиями работы брандмауера и предоставим основы знаний, необходимых для понимания основных принципов конфигурирования правил в pfSense.

Два осовных термина, которые будут наиболее часто встречаться по ходу этой части руководства, это "Правило" и "Набор правил". Правило - это конфигурация или действие для просмотра или управления трафиком. Набор правил - это сумма всех правил созданных пользователем или сформированных автоматически. В pfSense, как и в большинстве брандмауеров, наборы правил оцениваютя по принципу первого совпадения, что означает если Вы читаете список правил сверху-вниз - выполняться будет только первое правило удовлетворяющее условию. После достижения соответствия и выполнения действия требуемого правилом обработка останавливается. Об этом следует помнить всегда при создании нового набора правил, особенно если эти правила касаются ограничения трафика. Разрешающие правила должны быть расположены в нижней части набора, это позволит произвести исключения на ранних стадих фильтрации.

Размер таблицы состояний

Таблица состояний брандмауера имеет свой максимальный размер, сделано это для того, чтобы предотвратить исчерпание ресурсов памяти. Каждое такое состояние

занимает примерно 1Кб RAM. По умолчанию в pfSense размер таблицы состояний установлен в 10000, что значит что Вы можете иметь не более 10000 активных сетевых соединений проходящих через Ваш брандмауер, а любые дополнительные соединения будут отброшены. Это значение можно изменить на странице System -> Advanced в веб-конфигураторе (рис.1 "Увеличение размера таблицы состояний до 50000"). Введите требуемое число состояний и оставьте поле пустым, для того чтобы значение установилось по умолчанию.

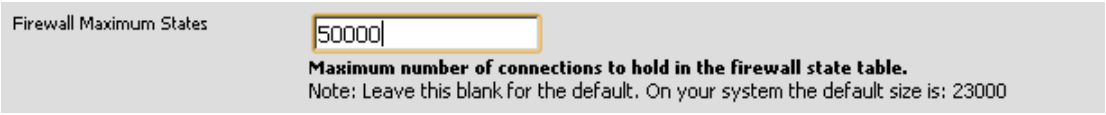


Рис.1. Увеличение размера таблицы состояний до 50000

Входная фильтрация

Входная фильтрация имеет отношение к трафику входящему в Вашу сеть из внешней сети Интернет. При использовании multi-WAN системы Вы получите несколько таких точек входа. По умолчанию в системе pfSense применяется политика блокирующая весь трафик приходящий на WAN извне. В свою очередь ответы WAN интерфейса на трафик инициируемый внутренними сетями позволяются автоматически посредством таблицы состояний.

Выходная фильтрация

Выходная фильтрация применяется к трафику инициируемому внутренними сетями, идущему на любой другой интерфейс брандмауера. В pfSense, как и в большинстве других брандмауеров по умолчанию применяется правило разрешающее весь трафик из LAN во внешнюю сеть.

Из своего опыта работы многочисленными брандмауэрами множества разработчиков во множестве организаций, я могу сказать, что большинство

пользователей не используют выходную фильтрацию. Это может вызвать административные проблемы, поскольку каждое новое приложение может потребовать дополнительных портов или протоколов в брандмауэре. в некоторых средах это весьма затруднительно, поскольку администраторы действительно не знают, что происходит в сети и опасаются повредить работе системы. В других системах, это не возможно по причинам политики рабочего места. Однако следует бороться за то, чтобы позволить покидать сеть минимальному необходимому трафику, насколько это возможно. Серьёзная выходная фильтрация важна по нескольким причинам:

1. Ограничить воздействие поставленной под угрозу системы -вредоносное ПО обычно использует порты не требуемые в стандартных сетях. Многие бот-вирусы используют порты для IRC, некоторые полагаются на более общие, такие как TCP 80 (HTTP) чтобы избежать выходной фильтрации. Например, если Вы запретите использование порта TCP 6667 (стандартный порт IRC), Вы сразу ликвидируете функционал множества ботов работающих по данному порту.

Вот характерный пример, который мы наблюдал - случай, когда внутренний интерфейс pfSense имел нагрузку 50-60 Мбит/с, в то время как WAN имел пропускную способность менее 1Мбит/с.

Некоторые случаи показывали, как боты используют систему LAN для организации DDoS атаки на различные сервера. Конкретно был использован порт UDP 80, видимо по ряду следующих причин. Во-первых, UDP позволяет отправлять большие пакеты, не завершая квитирования TCP. Для брандмауэров с сохранением состояний является нормой то, что большие пакеты TCP не будут передаваться, пока не будет успешно завершено квитирование, и это сильно ограничивает вероятность успешной DDoS атаки. Во-вторых, те кто использует выходную фильтрацию часто делают её слишком много допускающей, позволяя TCP и UDP там, где достаточно только TCP (например в случае HTTP).

В конкретном случае порт UDP 80 не был разрешён набором правил выходной фильтрации, и вся DDoS билась о внутренний интерфейс, отбрасываемая брандмауэром. Я случайно заметил такое поведение - брандмауэр продолжал пыхтеть и работать без

видимого ухудшения производительности, а администратор сети и не знал что происходит. Исходящий SMTP - другой характерный пример. Следует позволять SMTP только на порту TCP 25, что позволит отправлять почтовый трафик почтовому серверу вашей сети.

Если ваш почтовый сервер размещён внешне, позвольте вашей сети общаться с ним только по порту TCP 25. Это позволит исключить использование вашей системы в качесте спам-зомби. Вы внесёте свой вклад в ограничение спама и препятствуете тому, чтобы ваша сеть была добавлена в черный список Интернет. Корректное решение позволяет предотвратить такие проблемы, а выходная фильтрация обеспечивает другой уровень позволяющий ограничить воздействие различных проблем.

2. Предотвращение компроментации системы - в некоторых случаях выходная фильтрация может исключить компроментацию системы. Существует множество червей и различных эксплойтов, требующих исходящего доступа при попадании во внутреннюю сеть. Удачный пример такой ситуации - червь Code Red 2001 года, который использовал систему для получения вредоносного исполняющего файла по TFTP, а затем выполнял его.

3. Ограничение несанкционирования использования приложения - великое множество приложений, вроде VPN-клиентов, P2P-клиентов, различных мессенджеров, полагаются на нетиповые порты. Используя выходную фильтрацию, Вы можете эффективно и быстро ограничить несанкционированное использование таких приложений.

4. Предотвращение утечки информации - некоторым определенным протоколам вообще никогда нельзя позволять утечку из внутренней сети.Microsoft RPC на порту TCP 135, NetBIOS на портах TCP и UDP 137-139, а так же SMB/CIFS на портах TCP и UDP 445 - типичные порты служб которым нельзя позволять покидать пределы внутренней сети. Выходная фильтрация может предотвращать утечку информации о

нутренней сети в сеть Интернет и препятствует тому, что бы ваши системы инициировали попытки аутентификации с узлами в Интернет. Множество черверй полагалось на эти протоколы в недалёком прошлом.

Подходы к выходной фильтрации

Если в Вашей сети исторически долгое время не использовалась выходная фильтрация - может быть крайне сложне определить, какой трафик требуется для нормального функционирования. Данный раздел описывает некоторые подходы для реализации выходной фильтрации в вашей сети.

Первый из подходов - добавить разрешающие правила для трафика, в котором Вы уверены и готовы разрешить. Составьте список необходимых Вашей системе портов, из тех что Вы знаете, вроде таблицы похожей на 6.1. "Требования к исходящему трафику".

Требования к исходящему трафику Таблица 6.1.

Описание	IP источника	IP назначения	Порт назначения
HTTP и HTTPS со всех хостов	любой	любой	TCP 80 и 443
SMTP с почтового сервера	IP почтового сервера	любой	TCP 25
Рекурсивные запросы DNS с внутреннего DNS сервера	IP DNS сервера	любой	TCP и UDP 53

Затем, сконфигурируйте свои правила брандмауэра в соответствии с полученной таблицей и отбросьте всё остальное.

Другая альтернатива - включение журнала на ваши разрешающие правила и отправка журнальной (log) информации syslog серверу, для целей последующего анализа, который позволит проанализировать трафик покидающий вашу сеть. Два пакета анализа журналов поддерживают формат логов PF - fwanalog [1] и Hatchet [2]. Вы можете найти простые парсеры логов на основе сценариев, если имеете опыт работы с парсингом текстовых файлов. Это поможет создать необходимый набор правил с меньшим количеством ненужных остатков.

Блокировка против Отклонения. (Block vs Reject)

При создании правила существует два вида отклонения нежелательного трафика, это Блокировка (Block) и Отклонение (Reject). Блокировка реализует отклонение трафика в тихом режиме. Это поведение по умолчанию используется запрещающими правилами pfSense, следовательно в системе сконфигурированной по умолчанию, весь трафик со стороны внешней сети будет тихо отклонен. В свою очередь Отклонение (Reject) отправляет ответ на нежелаттельный TCP и UDP трафик, позволяя узлу, который инициирует данный трафик узнать, что в соединении ему отказано. Хотя можно установить Reject для любого правила брандмауэра, IP протоколы кроме TCP и UDP не могут быть отклонены - в любом случае будет произведена операция типа Block.

За прошедшие годы, среди профессионалов было много споров относительно сравнения использования Block и Reject. Некоторые утверждали, что использование Block имеет больше смысла, поскольку это "затрудняет" действия атакующих, сканирующих Интернет. Если вы используете Reject, ответ на сканирование закрытого порта сразу отсылается назад, в то время как использование Block тихо отбрасывает трафик, принуждая сканер атакующего ожидать ответ. Может быть это и так, однако, хороший сканер портов может сканировать одновременно сотни и тысяци узлов, и не дожидаться ответа от ваших закрытых портов.

Существует незначительное различие в потребление ресурсов и скорости сканирования, но настолько незначительное, что его можно просто не рассматривать. Если вы блокируете весь трафик из Интернет, проявляются значительные различия между работой Block и Reject - никто не определит, что ваша система находится в on-line режиме. Если есть хотябы один открытый порт, разница минимальна, поскольку атакующий узнает что вы в режиме on-line и узнает о наличии открытых портов и о том, что вы отклоняете блокированные соединения.

Поскольку нет существенного различия в блокировке через Reject, я рекомендую использовать Block для правил WAN. Для правил внутренних интерфейсов в большинстве ситуаций рекомендуется использовать Reject. Когда хост пытается получить

доступ к чему либо запрещённому правилами брандмауэра, приложение пытающееся поолучить доступ может зависнуть вплоть до получения ответа (или истечения внутреннего интервала ожидания соединения - п.п.).

При использовании Reject в соединение будет оказано немедленно, что позволит избежать зависания приложения. Обычно такое поведение может вызывать раздражение, но я обычно рекомендую использовать Reject, чтобы избежать возможных проблем сети при использовании Block.

Есть ещё один побочный эффект который может стать решающим фактором при выборе Block или Reject. Если вы используете Reject, это позволяет пользователям легко определить свои правила выходной фильтрации, поскольку брандмауэр сообщит им что блокируется. В принципе, для внутренних пользователей возможно разобраться в правилах выходной фильтрации и при использовании Block, правда это займёт немного больше времени и усилий.

Введение в экран настройки правил брандмауэра (Firewall Rules)

В этом разделе мы проведем краткий обзор страницы Firewall->Rules. В центральном списке Вы увидите набор правил WAN, который по умолчанию имеет лишь записи блокировки частных сетей (Block privete networks) и богон-сетей (Block bogon networks).

Рис.2. Правила WAN по умолчанию

Нажмите на закладку LAN, чтобы увидеть правила для LAN. По умолчанию, это только правило Default LAN -> any как видно на рис. 3. "Правила по умолчанию для LAN".

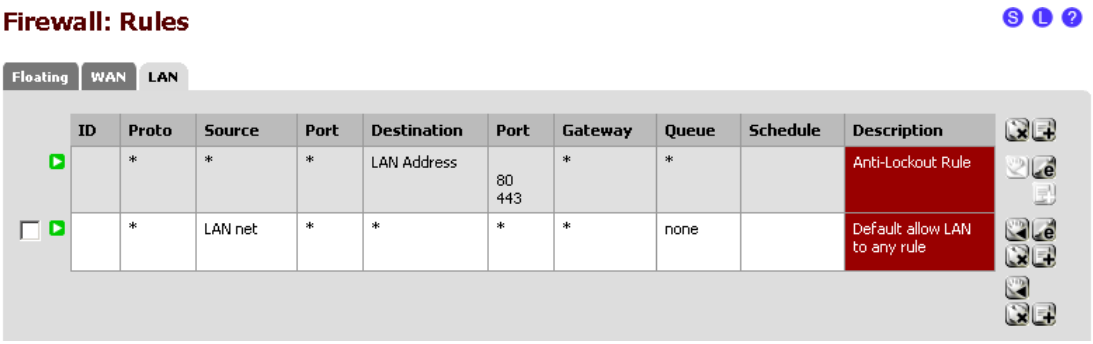


Рис. 6.3 Правила LAN по умолчанию.

Для каждого интерфейса в системе существует своя вкладка. Интерфейсы OPT будут отображаться своими описательными именами, т.е. если вы назвали свой интерфейс OPT1 как DMZ, то вкладка с его правилами будет называться DMZ.

Нажмите кнопку [+] на экране Firewall->Rules, чтобы добавить новое правило. Кнопки сверху и снизу, позволяют добавлять новые правила. Кнопка [+] сверху добавляет правило в верх набора правил, а кнопка [+] добавляет правило вниз списка. (рис.4. "Опции добавление правил LAN")

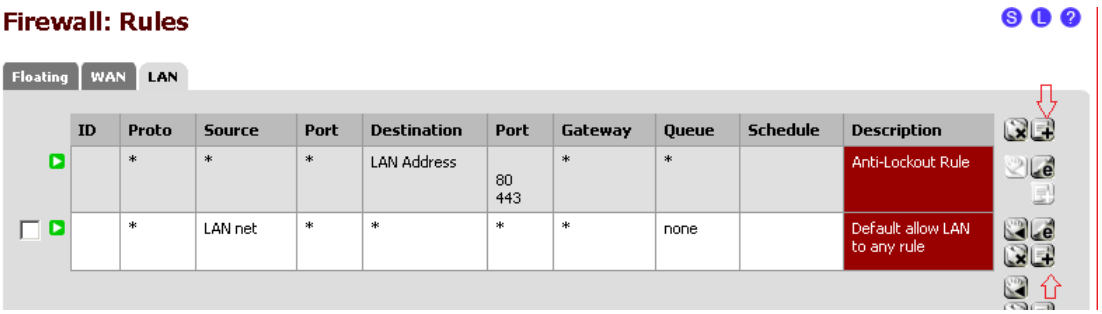


Рис 4. Опции добавление правил LAN

Чтобы изменить правило, нажмите кнопку [e] справа, или дважды щёлкните на строке правила. Вы попадёте на страницу, где сможете произвести изменения.

Правила могут быть перенесены в любую часть списка самостоятельно или в

группах. Чтобы сделать это, установите флажок на правилах которые должны быть перемещены, либо щёлкните по правилу, что так же приведёт к установке его флажка, затем нажмите кнопку [<] в строке, которая должна быть под перемещёнными правилами. Когда вы перемещаете указатель мыши, появится тусклая панель указывающая, куда будут вставлены правила. После щелчка, правила будут вставлены выше выбранной строки.

Для удаления правила, нажмите кнопку [x] справо. Вы получите запрос на подтверждение удаления. Для удаления сразу нескольких правил, установите флажки в начале строк тех правил, которые должны быть удалены и нажмите кнопку [x] в низу списка.

Алиасы

Алиасы помогут Вам сгруппировать порты, узлы, сети и обращаться к созданным группам при конфигурировании правил брандмауера, NAT и шейпера. Это позволит Вам создавать более эффективные и гибкие в настройке наборы правил.

Для создания нового алиаса, перейдите на страницу Firewall -> Aliases и нажмите кнопку [+]. В pfSense, каждый алиас ограничен 299 элементами (участниками). Для добавления новых элементов к алиасу нажмите [+] в нижней части списка записей на странице Firewall -> Aliases -> Edit.

Алиасы хостов позволяют создавать группы IP адресов. Рис. "Пример алиаса хостов" показывает пример использования алиаса хостов для создания списка публичных web серверов.

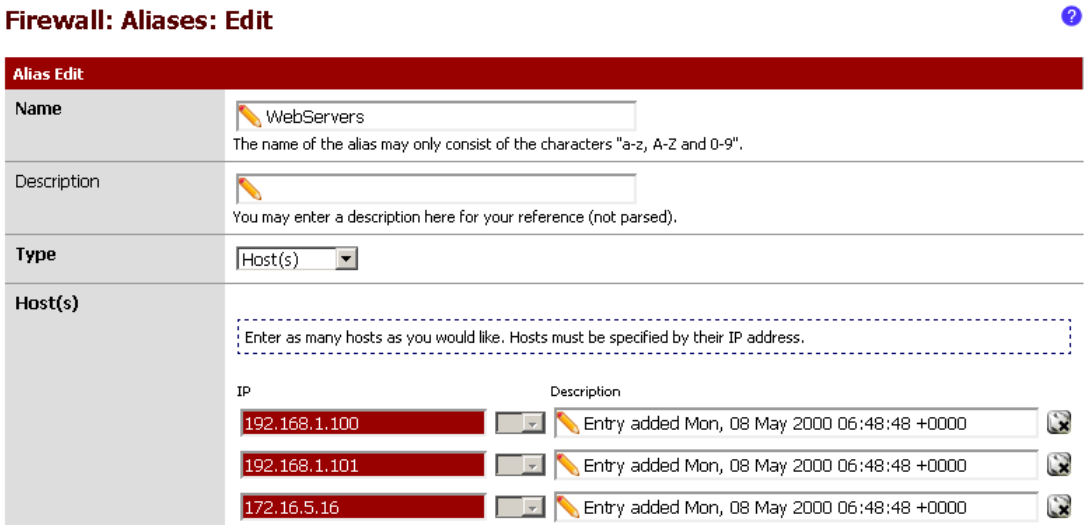


Рис. 5. "Пример алиаса хостов"

Алиасы сетей позволяют создавать группы сетей или диапазоны IP посредством использования CIDR суммаризации. Единичные хосты так же можно включить в сетевой алиас, выбрав /32 маску сети. Рис. 6. "Пример алиасов сети", показывает пример создания алиаса сети.

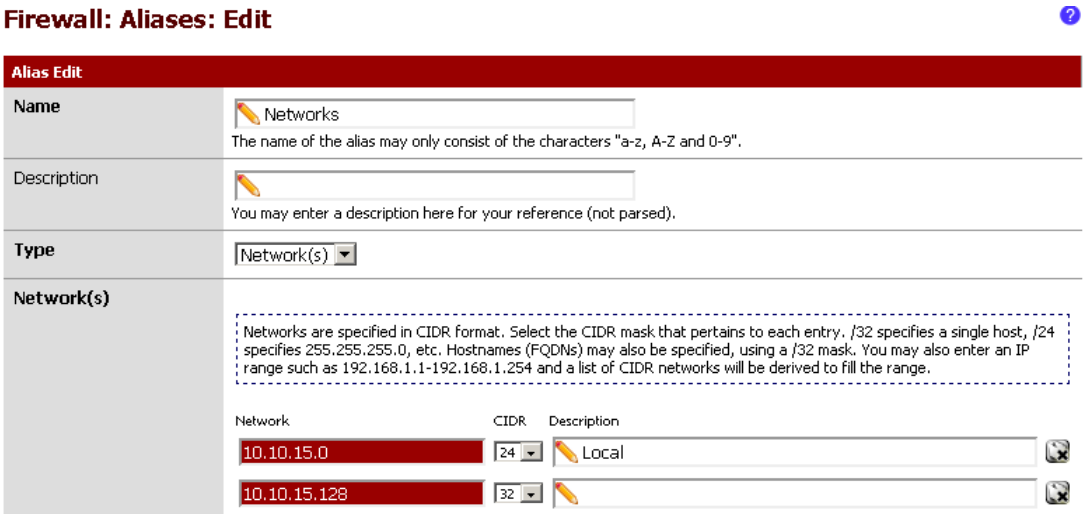


Рис. 6. "Пример алиасов сети"

Алиасы порта помогут произвести группировку портов и диапазонов портов. Протокол в алиасе не определяется, поскольку правило брандмауэра использующее алиас, определяет протокол. Рис.7 "Пример алиаса портов" показывает создание алиаса портов.

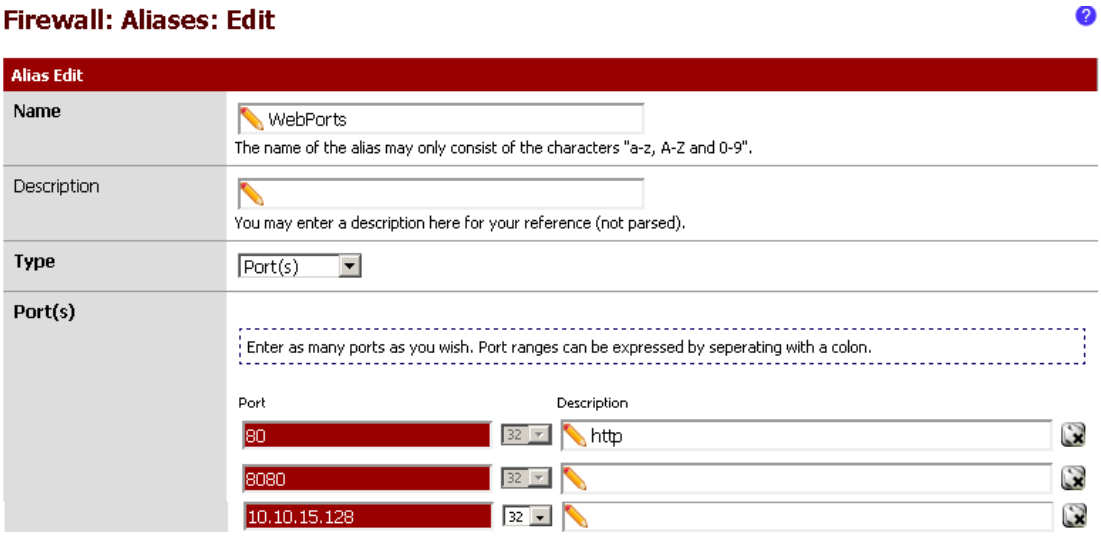


Рис. 7. "Пример алиаса портов"

В pfSense любое поле с красной подсветкой связано с алиасом. Если Вы введете в поле первую букву любого созданного алиаса - на экране появится выпадающий список соответствия, где Вы можете просто выбрать требуемый алиас.

Рис. 8. "Автозавершение алиаса хостов" показывает, как алиас WebServers сконфигурированный ранее может использоваться в поле Destination при добавлении или редактировании правил брандмауэра. Выберите "Single host or alias", и наберите первую букву требуемого алиаса. Показываются только алиасы соответствующего типа, например для поля, требующего IP адрес или подсеть, будут отображаться только алиасы хостов или сетей.

Рис. 8. Автозавершение алиасов портов

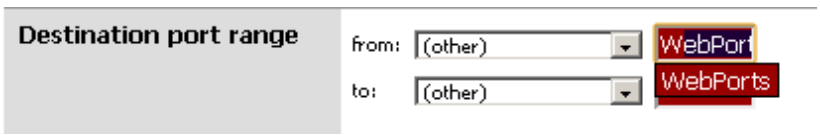


Рис. 9. Автозавершение для алиаса портов

Рис. 10. "Пример правила использующего алиасы" показывает правило, которое создано с использованием алиасов WebServers и WebPorts. Это правило находится на вкладке WAN и позволяет IP адрес любого источника, определённого в алиасе WebServers, который использует порты определённые в алиасе WebPorts.

Рис. 10 Пример правила использующего алиасы

Если вы наведёте курсор мыши на алиас в строке правила - появится поле с содержимым алиаса. Рис.11. показывает как это выглядит для алиаса WebServers, а рис.12 - соответственно для алиаса WebPorts.

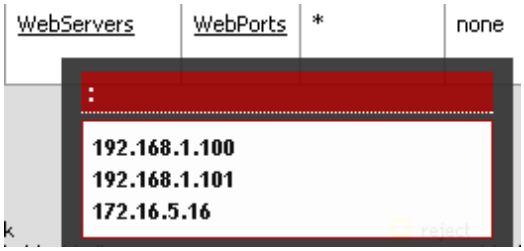


Рис. 11. Содержимое алиаса WebServers

Рис. 12. Содержимое алиаса WebPorts

По умолчанию ЗАПРЕЩЕНО

В сетевой безопасности существует два основных положения - по умолчанию РАЗРЕШЕНО (allow) и по умолчанию ЗАПРЕЩЕНО (Deny). Следует использовать стратегию по умолчанию ЗАПРЕЩЕНО для правил Вашего брандмауэра. Конфигурируйте свои правила на разрешение минимально необходимого трафика для потребностей вашей сети и отбрасывайте остальное. В следствии этой методики, число запрещающих правил будет минимальным. В конфигурации по умолчанию, pfSense использует на WAN интерфейсе правило по умолчанию ЗАПРЕЩАТЬ, а на LAN - РАЗРЕШАТЬ.

Весь входящий трафик Интернет запрещается, а весь трафик от LAN в интернет разрешается. Все маршрутизаторы домашнего использования используют эту методику, так же как и все проекты с открытым исходным кодом и аналогичные коммерческие решения. Однако, это не рекомендуемые параметры для работы. Пользователи pfSense часто спрашивают, "что я должен блокировать?". Такой вопрос не корректен, поскольку применим только к методике РАЗРЕШАТЬ ВСЁ по умолчанию.

Чем короче Ваш набор правил, тем легче им управлять. Длинные наборы увеличивают вероятность ошибки, имеют склонность к чрезмерным разрешениям и трудно контролируемы. Используйте алиасы это поможет Вам максимально сократить ваши наборы правил.

Сокращение журнала

По умолчанию в pfSense включено журналирование (log) для правила ЗАПРЕЩЕНИЯ. Это означает, что все попытки соединения которые будут заблокированы заносятся в системный журнал. Иногда лишней информации не много, но в большинстве сред журналы сильно разростаются. У провайдеров кабельного интернета - журналы забиваются сообщениями NetBIOS от Windows машин напрямую соединёнными с широкополосными каналами. Эти хосты постоянно формируют широковещательные запросы при просмотре сети. Весь этот мусор может скрывать сообщения которые действительно могут быть важными. При добавлении на WAN интерфейс блокирующего правила без включения журналирования, этот трафик будет блокироваться, но больше не

будет журналироваться.

Рис. 13. Правило брандмауэра для предотвращения журналирования широковещательных сообщений

Следует добавлять подобные правила в соответствии со специфическими особенностями журнального мусора, который вы можете наблюдать в своей среде. Проверьте состояние журналов брандмауэра на странице System Logs -> Firewall, чтобы видеть, какой трафик вы блокируете и частоту данного трафика. Если какой-то трафик упоминается более чем 5 раз в минуту, вероятно следует добавить соответствующее правило блокировки, чтобы снизить лишний журнальный шум.

После установки pfSense не журналирует любой разрешённый трафик и журналирует весь отклоненный трафик. Это стандартное поведение по умолчанию практически любого брандмауэра. Такое поведение наиболее практично.

Методика создания правил

В pfSense правила брандмауера применяются на базе понятия интерфейса, всегда в направлении входящего трафика. Это означает, что трафик, приходящий на LAN, фильтруется с использованием правил интерфейса LAN итд. И поскольку все правила созданные в pfSense, по умолчанию имеющие сохранение состояний - весь трафик разрешенный определенным правилом создает соответствующую запись в таблице состояний. Весь ответный трафик разрешается автоматически, в соответствии с данной записью. В некоторых случаях pfSense автоматически добавляет некоторые правила брандмауэра. В этом разделе мы рассмотрим каждое из автоматически создаваемых правил и цель их создания.

Что бы предотвратить блокировку web интерфейса, pfSense по умолчанию

включает правило антилокаута. Это поведение конфигурируется на странице System->Advanced WebGUI Anti-lockout. Данное автоматическое правило разрешает трафик из любого источника вашей сети к любому протоколу слушающему на LAN IP. В средах осознающих безопасность, следует отключить данное правило и конфигурировать правила LAN так, чтобы только алиас доверенных хостов мог получать доступ к административным интерфейсам брандмауэра.

Так же pfSense использует функцию антиспуффинга. Эта функция реализует функциональность uRPF (Unicast Reverse Path Forwarding) в соответствии с RFC 3704. Брандмауэр проверяет каждый пакет в соответствии со своей таблицей маршрутизации и если попытка соединения происходит с исходного IP на интерфейсе, где по данным брандмауэра такой сети нет - пакет отбрасывается. Например, нечто приходящее с WAN интерфейса с адресом внутренней сети будет отбрасываться. Так же, что либо инициируемое из внутренней сети с IP источника который не находится в этом внутреннем сегменте тоже будет отбрасываться.

Опция блокировки частных сетей на WAN интерфейсе автоматически формирует правило для подсети RFC 1918. Если у вас нет частного пространства IP адресов на вашем WAN, эту опию следует включить. Опция применяется только к трафику инициируемому на стороне WAN. Из внутренней сети всё ещё возможно получить доступ к узлам частных сетей.

Bogon networks - это сети, которые никогда не должны наблюдаться в Интернет, включая зарезервированное и не распределённое пространство IP адресов. Эти сети никогда не должны присутствовать как источник в Интернет, а их наличие указывает на спуффинг или на "угнанную" сеть используемую в злонамеренных целях. В pfSense реализован список богон-сетей, который обновляется ежемесячно. Если у вас ключена блокировка богон-сетей, ваш брандмауэр будет получать обновления списка богонов с files.pfsense.org в первый день каждого месяца. Следует убедиться, что брандмауэр разрешает DNS имена хостов, иначе обновление не будет работать. Для проверки работы DNS, перейдите на страницу Diagnostics->Ping и запустите ping на files.pfsense.org как показано на рис.20 "Тестирование разрешение имён для обновления списка

богонов".

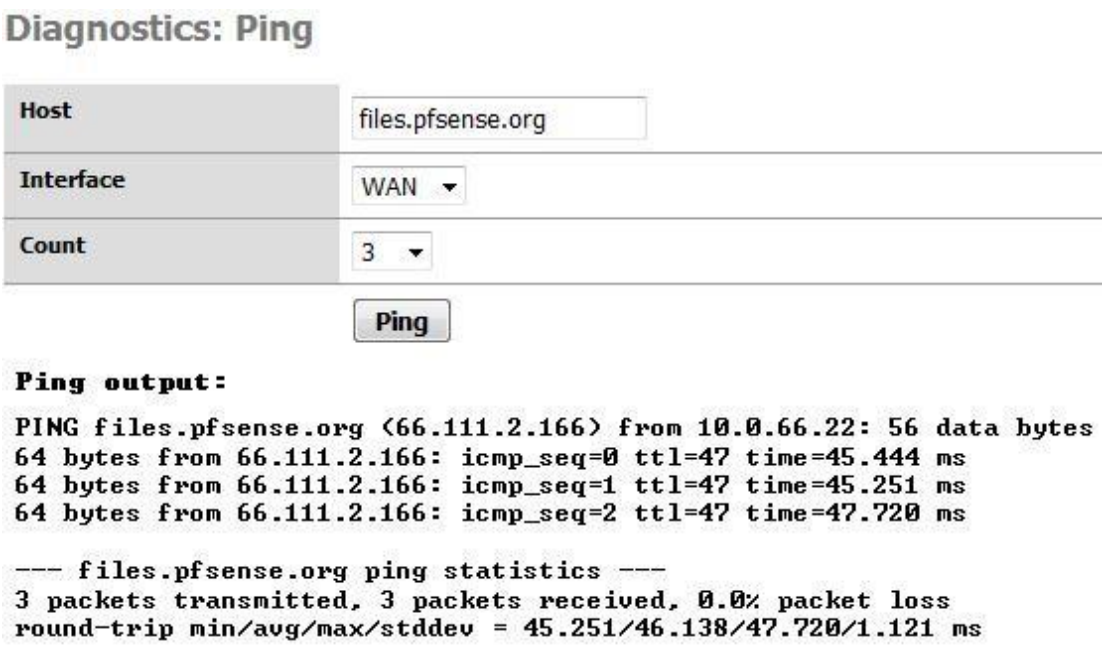


Рис. 20. Тестирование разрешение имён для обновления списка богонов

Если у Вас все же наблюдаются проблемы с разрешением DNS-имен, можно вручную выполнить обновление через страницу Diagnostics -> Command, выполнив

#/etc/rc.update_bogons.sh now.

Аргумент **now** следующий за именем скрипта сообщает о необходимости немедленного выполнения скрипта.

Когда вы подключаете IPSec соединение, автоматически добавляется правило позволяющее удалённый тунель доступа IP адреса конечной точки к порту UDP 500 и протокол ESP на WAN IP используемом для соединения. Когда мобильные клиенты IPSec включены, трафик UDP порт 500 и протокол ESP разрешаются с любого источника. Из-за способа работы политики маршрутизации, любой трафик который соответсвует правилу определяющему шлюз будет вытеснен к Интернет и обойдёт обработку IPSec. Когда у вас есть разрешающее правило определяющее шлюз на внутреннем интерфейсе содержащем подсеть используемую IPSec соединением, и место назначения - any (любое), правило добавляется автоматически дабы инвертировать политики маршрутизации для трафика предназначенного удалённой подсети VPN. Автоматически

добавляемые правила для IPSec обсуждаются более развернуто в Части 13 данного руководства.

Когда вы подключаете PPTP сервер, автоматически добавляются скрытые правила позволяющие порт 1723 TCP и протокол GRE (Generic Routing Encapsulation) к вашему WAN IP адресу с любого исходного IP.

Правила, которые не соответствуют любым правилам определённым пользователем или любым другим автоматически добавленным правилам будут тихо блокироваться правилом запрета по умолчанию.

Виртуальные IP

pfSense позволяет использовать множественные публичные IP адреса в соединение с NAT через виртуальные IPs (VIPs). Существует три типа виртуальных IP доступных в pfSense: Proxy ARP, CARP и Other. Каждый из них полезен в различных ситуациях. В большинстве случаев pfSense необходимо обеспечить ARP на ваших VIPs, следовательно, следует использовать Proxy ARP или CARP. В ситуациях где ARP не требуется, например когда дополнительные публичные IP маршрутизируются вашим провайдером к вашему WAN IP, используйте тип Other VIPs.

Proxy ARP функционирует строго на уровне 2, просто обеспечивая ответы ARP для указанного IP-адреса или диапазоан CIDR IP адресов. Это позволяет pfSense передавать трафик, предназначенный этому адресу согласно вашей конфигурации NAT. Адрес или диапазон адресов не присваивается никакому интерфейсу pfSense. Это означает, что никакие службы самого pfSense не могут

отвечать на этих IP адресах. Обычно это считается преймуществом, поскольку ваши дополнительные IP адреса должны использоваться только в целях NAT.

VIP CARP в основном используется при избыточном развёртывании использующим CARP. Для получения информации об использовании VIP CARP смотрите Часть 20 руководства.

"Other" VIP позволяют определять дополнительные IP адреса в случае их использования когда ответ ARP не требуется. Единственная функция добавления Other VIP делает этот адрес доступным на экране конфигурации NAT. Это бывает полезно, когда у вас имеется блок публичных IP, маршрутизируемый к вашему IP адресу WAN или VIP CARP.

Правила основанные на времени (Time Based Rules - TBR)

TBR поможет Вам использовать любые правила брандмауера только в определенные дни или временные промежутки. Но в тоже время есть некоторые противопоказания при использовании TBR. В данном разделе мы расскажем, как наиболее правильно использовать TBR и чем же они отличаются от остальных правил.

Планировщик TBR определяет время, когда необходимо выполнить действие описаное в правиле, а если текущее время или дата не включены в расписание - действие правила инвертируется. Например, если правилом указано разрешать трафик с определенного узла только по субботам - во все остальные дни, этот трафик будет автоматически блокироваться, вне зависимости от любых других правил. Применяется первое соответствие и как только соответствие обнаруживается, предпринимаются соответствующие процедуры, дальнейшие правила уже не оцениваются. Если Вы

работаете с разрешающим правилом по определённому расписанию, например в субботу и воскресенье, не имеющего целевого эффекта в эти дни, Вы можете вместо этого использовать блокирующее правило с понедельника по пятницу. Всегда важно помнить, используя расписание, что правило будет иметь некотрый эффект, является это в рамках запланированного правила или нет.

Так как правила TBR основаны на ipfw, а не на pf, как все остальные правила pfSense - они не совместимы в работе с каптивным порталом. По этой же причине, некоторые усовершенствованные функции брандмауера, вроде multi-WAN - тоже не доступны при использовании TBR.

Создать расписание можно на странице Firewall -> Schedules, при этом в каждое расписание могут быть включены множественные временные промежутки. После определения расписания его можно использовать при создании правил брандмауера. В слудеющем примере мы рассотрим способ ограничить доступ к определенному HTTP во время рабочего дня и открыть доступ в отсальное время

Чтобы создать новое расписние щелкните [+] на странице Firewall -> Schedules. Вы перейдёте на экран редактирования расписания, как показано на рис.23. "Добавление диапазона времени". Первое поле на этом экране - Schedule Name (Имя расписания). Это то имя которое появится в списке выбора при использовании в правилах брандмауэра. Как и имена алиасов, это имя должно содержть буквы и цифры без пробелов. Для нашего примера используем имя BussinesHours. Затем, в поле Description (Описание), введите более подробное описание расписания в свободной форме, например Standart Business Hours.

Поскольку расписание составляется из одного или более диапазонов, следует прежде определить диапазон времени, а затем сохранить расписание. Расписание может применяться к определённым дням, например 2 сенятбря 2009, или к дням недели,

например Monday-Wednesday. Для выбора любого дня в году выберите из выпадающего списка месяц, затем щёлкните по соответствующему дню или дням на календаре.

Для выбора дня недели щёлкните по её имеи в заголовке столбца. Для нашего примера щёлкнем по понедельнику, вторнику, среде, четвергу и пятнице. Расписание станет активным в любой понедельник-пятницу независимо от месяца. Теперь выберем время, в которое это расписание будет активным в формате 24 часов. Наше рабочее время 9:00 - 17:00 (5pm). Выбор производится в зоне местного времени. Введите описание диапазона времени (Time Range Description), например Work week, и нажмите Add Time (Добавить время).

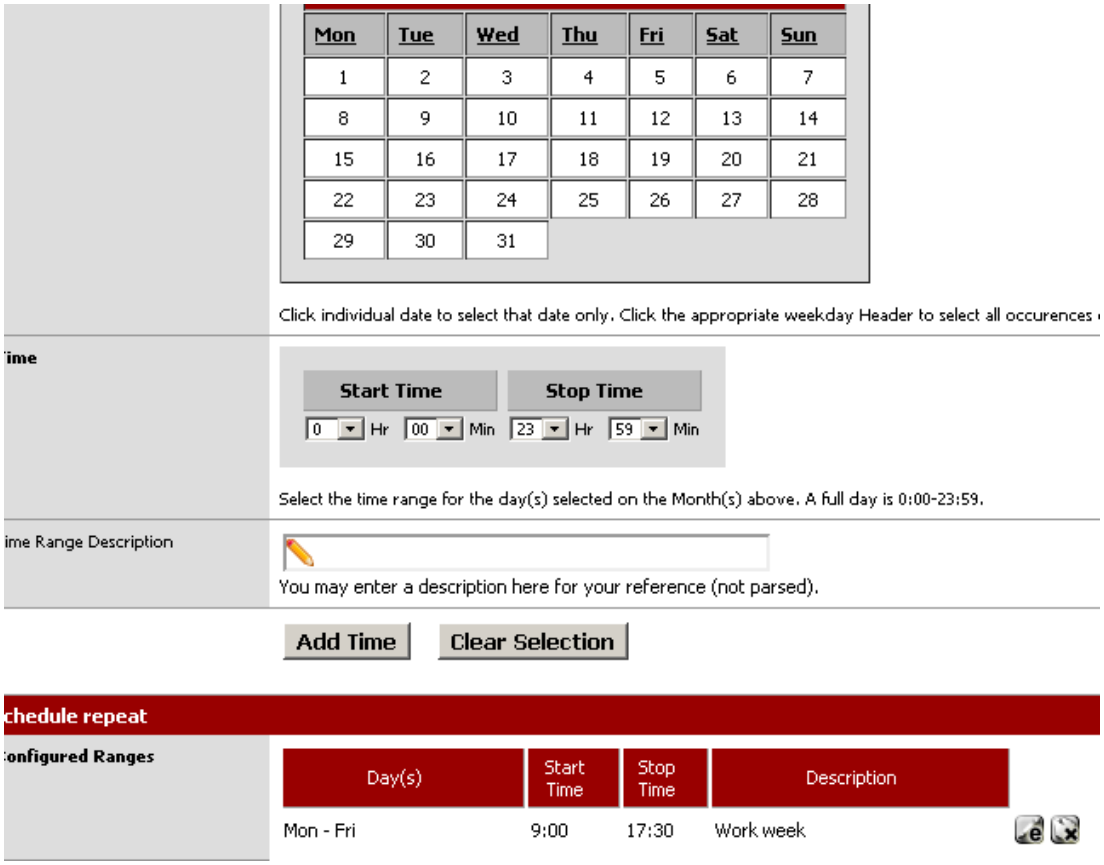


Рис. 23. "Добавление диапазона времени".

Если необходимо, повторите процесс пока не получите требуемый результат. Когда все необходимые диапазоны времени будут определены нажмите Save. Вы возвратитесь к списку расписаний и новое расписание будет отображено как показано на рис.25. "Список расписаний после добавления". Теперь это расписание доступно для использования в правилах брандмауэра.

Рис.25. "Список расписаний после добавления".

Чтобы создать правило брандмауэра использующее данное расписание следует добавить правило на требуемом интерфейсе. "Конфигурирование правил брандмауэра" для получения более подробной информации о добавлении и редактировании правил. Для нашего примера добавьте правило блокировки трафика TCP для интерфейса LAN из подсети LAN, к любому месту назначения на порту HTTP. Когда дойдёте до установки Schedule, выберите расписание BussinesHours, которое мы только что определили, как показано на рис.26. "Выбор расписания для правила брандмауэра".

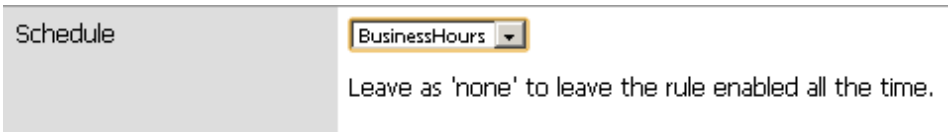


Рис. 26. "Выбор расписания для правила брандмауэра"

После сохрания правила, оно появится в списке правил брандмауэра, так же Вы увидите индикацию о том, что расписание включено. Как видно из рис.27. "Список правил брандмауэра с расписанием" - это блокирующее правило, но столбец расписания указывает, что в настоящий момент правило находится в неактивном состоянии, поскольку диапазон текущего времени находится за пределами расписания. Если навести мышь на имя расписания -будет показан диапазон времени определённый для данного расписания.

Рис. 27. "Список правил брандмауэра с расписанием"

Просмотр журналов брандмауэра

Для всех запрещающих правил, а также для тех в настройках которых была установлена отметка о логировании - производится запись в журнал. Есть много способов просмотра этих записей и нельзя однозначно говорить о "лучшем методе" просмотра. Как и все остальные журналы pfSense, логи брандмауера имеют ограничение по количеству записей.

Журналы брандмауера можно просмотреть в web-интерфейсе pfSense на странице Status -> System Logs, вкладке Firewall. Вы можете просматривать уже отфильтрованные логи или работать с "сырыми журналами" включающими более подробную информацию, если понимаете формат журналирования PF. Парсинг журнала черех web-интерфейс показан на рис.28. "Пример записей журнала WebGUI". Можно видеть шесть столбцов:

Action (Действие), Time (Время), Interface (Интерфейс), Source (Источник), Destinanion (Назначение) и Protocol (Протокол).

- - Action, показывает что произошло с пакетом который сгенерировал запись журнала, был ли он пропущен, заблокирован или сброшен.
- - Time, время прибытия пакета
- - Interface, интерфейс на котором находился пакет
- - Source, исходный IP адрес и порт.
- - Destination, целевой IP адрес и порт.
- - Protocol, протокол пакета, TCP, UDP, ICMP и пр.

Act	Time	If	Source	Destination	Proto
⊠	Jul 16 20:54:05	WAN	0.0.0.0:68	255.255.255.255:67	UDP
⊠	Jul 16 20:56:05	WAN	0.0.0.0:68	255.255.255.255:67	UDP
⊠	Jul 16 21:05:05	WAN	0.0.0.0:68	255.255.255.255:67	UDP
⊠	Jul 16 21:06:05	WAN	0.0.0.0:68	255.255.255.255:67	UDP

Рис. 28. "Пример записей журнала WebGUI"

По нажатию на значок действия, Вы перейдете по ссылке которая приведет Вас к правилу вызвавшему эту запись журнала. Таким способом можно быстро диагностировать суть проблемы. Если был использован протокол TCP, то Вы увидите дополнительные поля, которые отображают флаги TCP присутствующие в пакете. Вот перечень наиболее общих флагов:

S - SYN Синхронизация порядковых номеров. Когда устанавливается только SYN - указывает на попытку нового соединения.

A - ACK указывает на подтверждение данных. Как говорилось ранее, это позволяет отправителю узнать что данные были нормально получены.

F - FIN указывает, что больше данных от отправителя нет и закрывает соединение.

R - RST сбрасывает соединение. Этот флаг устанавливается в ответ на запрос открытия соединения на порту не имеющему слушающего демона. Так же может устанавливаться брандмауэром для отклонения нежелательных соединений. Кроме того существуют другие флаги, значеие которых можно получить в документации по протоколу TCP. Следует обратиться к расширенным источникам.

Просмотр RAW журналов.

"Сырые" или raw журналы можно просматривать в режиме реального времени из консольного меню при выборе пунка 10. Простой пример - запись журнала показаного на рисунке 6.28:

000000 rule 54/0(match): blockinonvr1: 0.0.0.0.68 > 255.255.255.255.67: BOOTP/

Запись показывает что было использовано 54е правило приведшее к действию блокировки на интерфейсе vr1. Затем показываются IP адреса источника и назначения. Пакеты других протоколов могут предоставить значительно больший объём информации.

Журнал фильтра, как мы уже говорили в начале главы, содержит бинарный циркулярный журнал, следовательно невозможно использовать традиционные инструметы, подобные cat, grep и пр. для непосредственной работы с файлом. Журнал

должен считываться с использованием clog и только тогда может передаваться с использованием канала в требуемую программу на обработку.

Для того чтобы "следовать" за выводом clog файла, Вы должны использовать параметр clog -f. Это эквивалентно tail -f при работе с обычными файлами системных журналов UNIX. Будет выведено всё содержимое файла системного журнала без выхода. Команда будет ожидать новых записей и выводить их на экран по их поступлению в журнал.

Есть простой синтаксический анализатор журналов, написаный на PHP, который может использоваться из оболочки для вывода упрощённого вида вместо сырого. Чтобы просмотреть такой парсинг журнала используйте следующую команду:

#clog /var/log/filter.log | php /usr/local/www/filterparser.php

Вы увидите записи журнала в выводе следующего вида:

Jul 17 00:06:05 block vr1 UDP 0.0.0.0:68 255.255.255.255:67

Просматривая сырой журнал, Вы также видете номер правила. Можно использовать этот номер для поиска правила вызвавшего запись. В следующем примере мы попытаемся узнать, какое правило идёт под номером 54:

#pfctl -vvsr | grep '^@54'

@54 block drop in log quick all label "Default deny rule"

Как можно видеть - это правило блокировки по умолчанию.

Иногда Вы будете видеть записи журнала, которые отмечены правилом блокироки по умолчанию, хотя по всем признакам они принадлежат разрешённому трафику. Наиболее частый пример - подключение web сервера. Вероятно, это происходит когда пакеты FIN TCP, который обычно закрывает соединение, прибывает после того как состояние соединения было удалено. Пакет будет потерян и передача будет блокирована поскольку брандмауэр уже закрыл соединение. Так поступают все stateful брандмауэры, хотя некотрые из них не генерируют запись журнала для такого блокированного трафика, даже когда вы регистрируете весь блокированный трафик.

Поиск и устранение неисправностей в правилах брандмауэра

В этом разделе мы рассмотрим действия которые необходимо проветси, если Ваши правила ведут себя иначе, нежели Вы хотите.

Первым делом стоит проверить записи журналов брандмауера (Status -> System Logs на вкладке Firewall). Нужно помнить, что pfSense не регистрирет пропущенный трафик и регистрирует весь блокированный. Если при просмотре журнала Вы не видите трафика отмеченного красным "X", это означает, что pfSense не отбрасывает трафик.

Отредактируйте рассматриваемое правило и изучите параметры, которые вы определили для каждого поля. Для TCP и/или UDP трафика следует помнить, что исходный порт почти никогда не совпадает с портом назначения и должен устанавливаться в any. Если значение по умолчанию - блокировка, возможно вам придётся создать новое правило соответствующее разрешённому трафику.

Важно также всегда помнить главное - после первого совпадения дальнешие правила не оцениваются!

Стоит убедиться, что Ваши правила находятся на верном интерфейсе брандмауера. Трафик генерируемый на LAN и идущий на другие интерфейсы системы, фильтруется только по правилам на самом LAN интерфейсе. Это справедливо ко всем интерфейсам системы.

Может быть полезно определить, какому правилу соответствует рассматриваемый трафик. Включив журналирование проходящего трафика вы можете видеть журналы брандмауэра и выбирать отдельные записи для просмотра правила передавшего трафик.

Еще одна функция "Захват пакетов" может стать неоценимой помощью для поиска и устранения проблем. Таким способом можно определить всю информацию о пакете: достигает ли трафик внешнего интерфейса в принципе, покидает ли трафик внутренний интерфейс и многое другое. Чтобы получить больий объем информации о поиске и устранении проблем с помощью захвата пакетов и tcpdump, читайте Часть 25 данного руководства "Захват пакетов".

Виртуальный сетевой экран

Все прелести использование виртуального файрвола изначально были изложены еще в 2005 году Василисом Превалакисом, профессором зарубежного университета (Drexel University, Philadelphia http://www.prevelakis.net/Papers/VirtualFirewall.pdf).

Наш бывший соотечественник предлагал использовать в качестве виртуального файрвола PF и OpenBSD 3.7, работающей при помощи бесплатного VMPlayer.

Pf был портирован и удачно работает во FreeBSD, а на горизонтах виртуализации стабильно развивается проект от Sun Microsystems под названием VirtualBox. Их как раз и будем использовать. Был взят дистрибутив PfSense, включающий в себя непосредственно FreeBSD и уже минимально настроенный файрвол. Плюс ко всему возможность настройки маршрутизатора через веб-интерфейс, что несомненно облегчит жизнь 95% пользователей.

Виртуальная машина выступает в роли маршрутизатора и файрвола для Вашего основного компьютера. Попадая на физический сетевой интерфейс, пакет сразу передается в виртуальное окружение, проходит обработку на "пригодность" и в случае положительного заключения передается через виртуальный интерфейс Вам. Для Вас это делается абсолютно прозрачно и Вы не видите разницы.

Подготовка к работе

Имеем Windows 7, интернет. Необходимо скачать последний **VirtualBox** и **PFsense** – дистрибутив FreeBSD в виде LiveCD , в который включен портированный из OpenBSD сетевой экран pf.

Порядок действий

Качаем, ставим VirtualBox. Запускаем его, создаем новую машину pfsense с динамическим диском 2 Гб, 128 Мб памяти, как на рис.29.

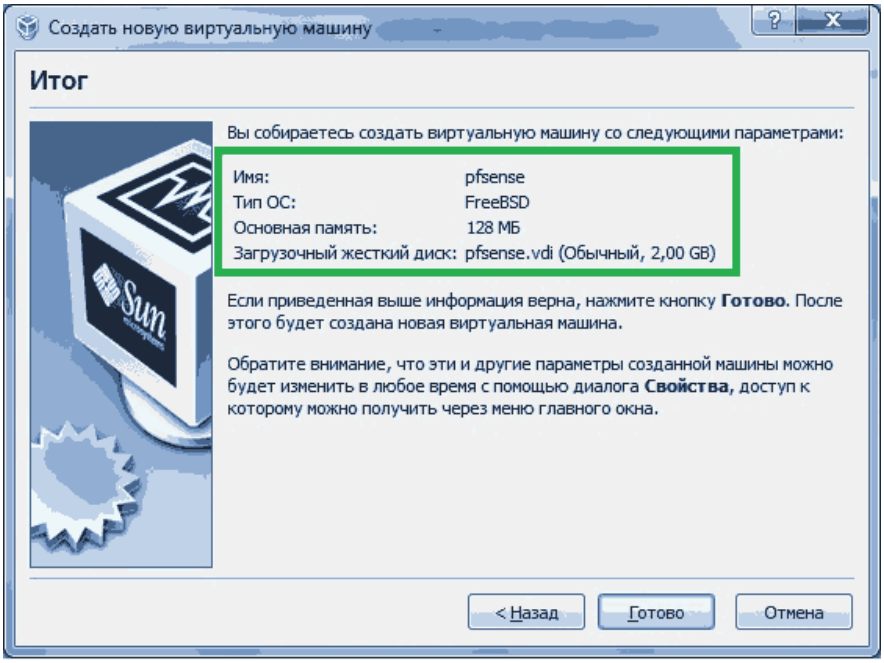


Рис.29 настройка виртуальной машины

После создания не торопимся её запускать. Настраиваем сетевые интерфейсы. Первый – внешний (WAN), он включается мостом к реальному сетевому адаптеру.

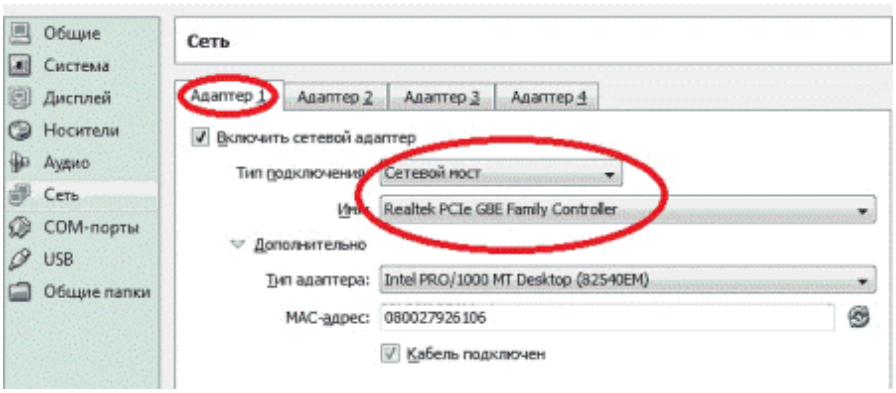


Рис.30 Настройка первого сетевого интерфейса

Второй – внутренний (LAN), направлен на виртуальный адаптер хоста.

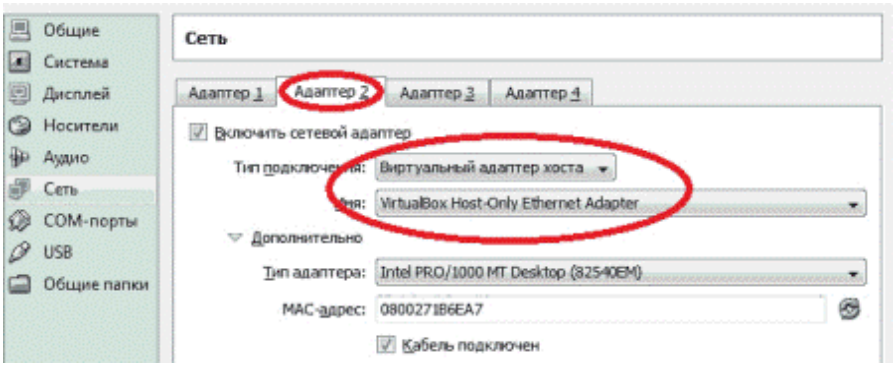


Рис.31 Настройка второго сетевого интерфейса

При этом необходимо записать последние четыре цифры mac-адреса каждого сетевого интерфейса. В нашем случае – WAN=6106, а LAN=6EA7. Последним штрихом подключаем образ pfsense.iso к виртуальному CD.

Рис.31 Монтирование виртуального CD-ROM

Теперь надо настроить реальные интерфейсы хоста. Пуск-Панель управления-Центр управления сетями и общим доступом. Там выбираем изменение параметров адаптера. Видим два (как минимум, нужных нам) адаптера.

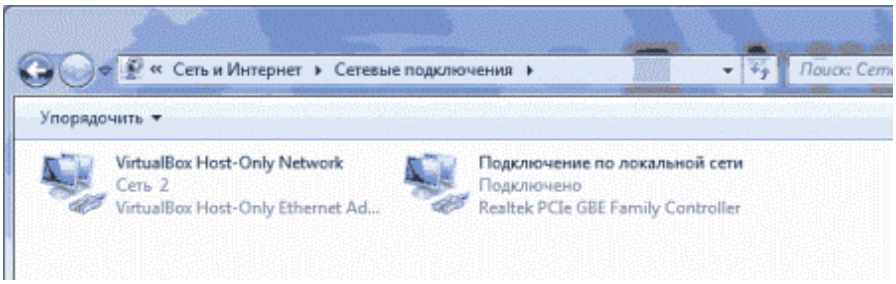


Рис.32 Появление виртуального сетевого интерфейса в компьютере

Для реального адаптера (в моем случае – это Realtek) отключаем все, кроме VirtualBox драйвера,

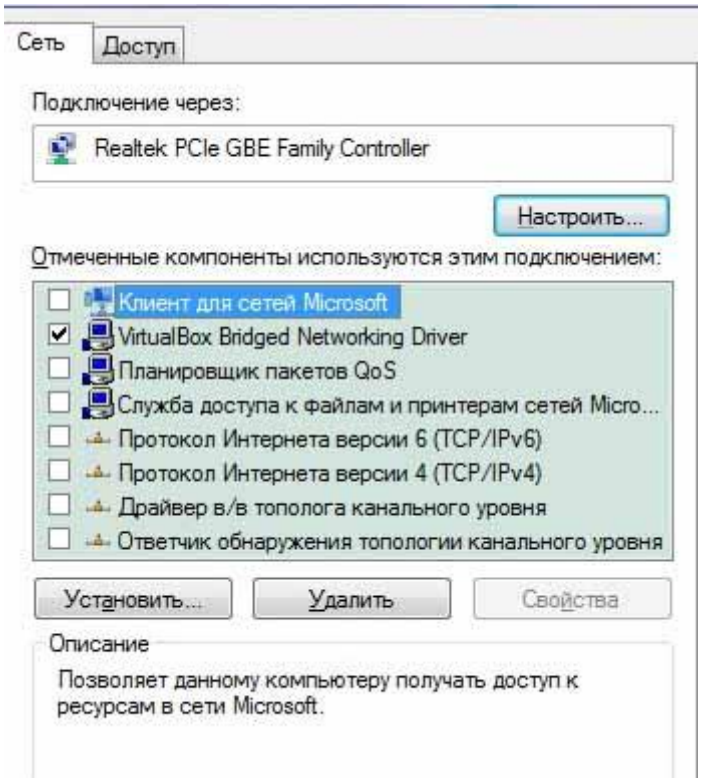


Рис.33 Настройка реального сетевого интерфейса

а для виртуального адаптера (VirtualBox Host-Only Network) оставляем все,

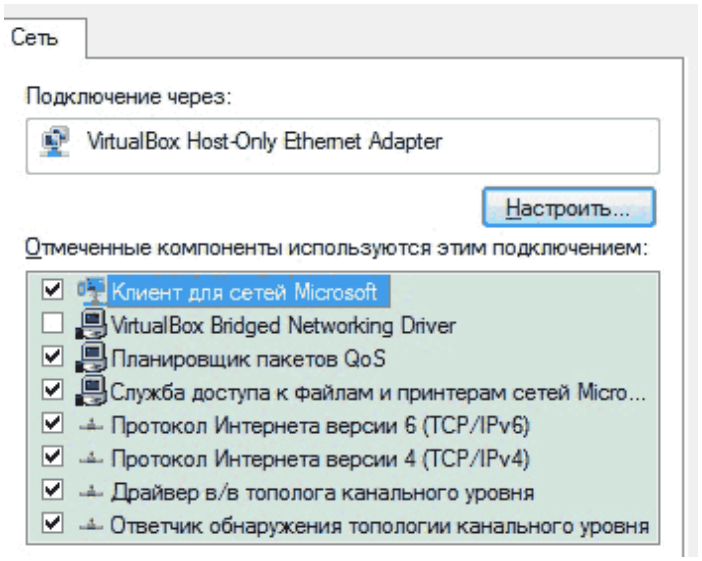


Рис.34 Настройка виртуального сетевого интерфейса

только донастраиваем TCP протокол так, чтобы наш компьютер для выхода в сеть использовал виртуальную машину и как шлюз, и как DNS сервер.

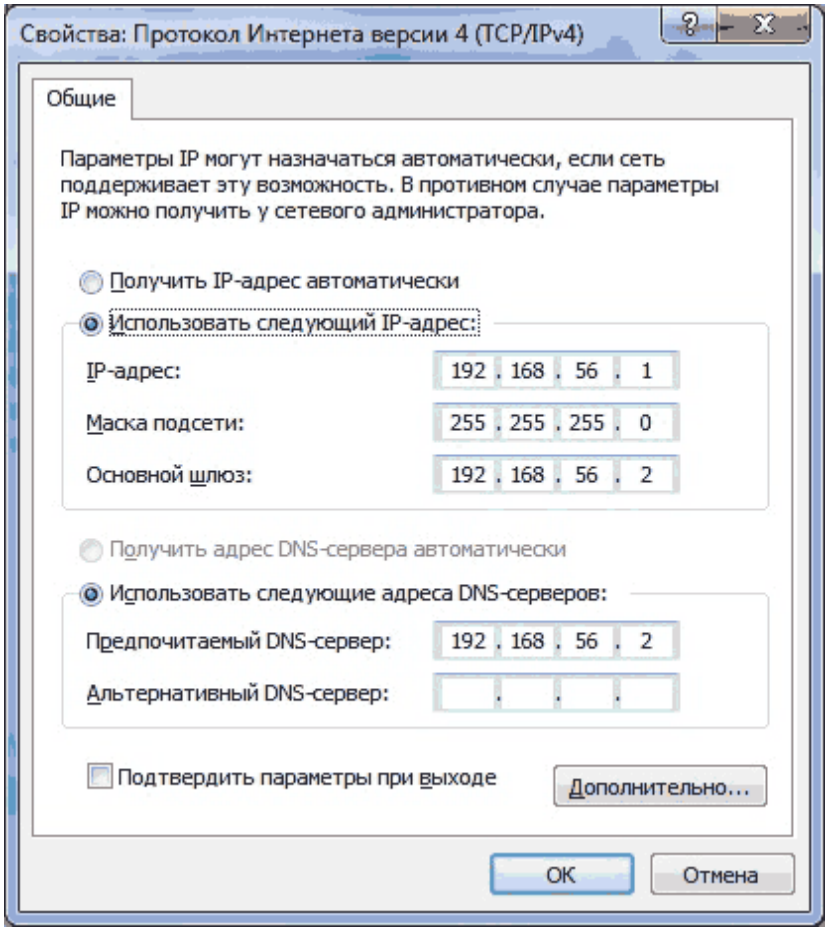


Рис.35 Настройка протокола интернета

Например, в моем случае, для интерфейса компьютера я оставил адрес 192.168.56.1, а для локального интерфейса PFsense – 192.168.56.2, который станет для меня и шлюзом и DNS сервером.

Внимание!!! После этих настроек у Вас пропадет интернет до окончания установки и настройки PFsense.

Для ускорения запуска виртуальной машины можно отключить FDD привод, Аудио драйвер, все COM и USB порты.

Вот теперь можно запустить машинку. До появления вопроса о VLAN ничего не делаем. Про VLAN отвечаем n (нет).

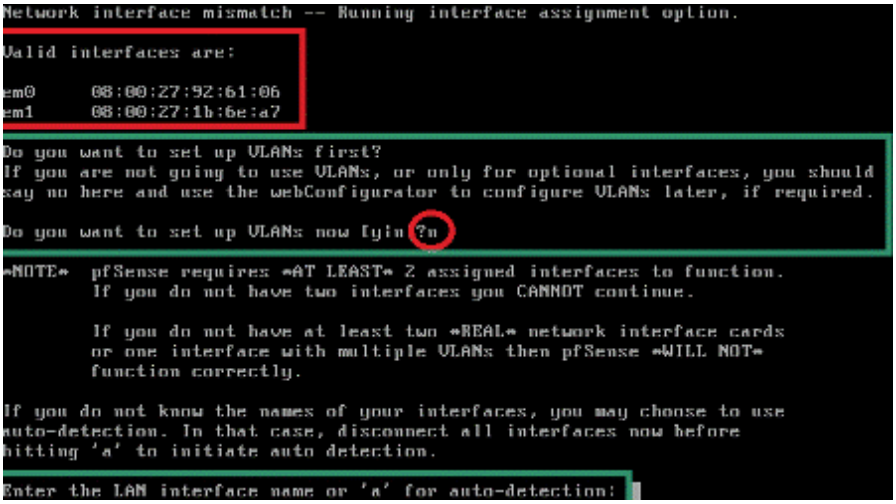


Рис.36 Запуск установки pfSence, предварительная настройка

На этом же рисунке мы видим несколько интересных моментов. Красной рамкой обведены сетевые интерфейсы, которые определила система, и мы видим, что WAN=em0 (61-06), а LAN=em1 (6E-A7)**.** В зеленой рамке, как уже упоминалось- отказ от настройки VLANa, а ниже (в синенькой) вопрос о соответствии интерфейсов. Как раз здесь-то и надо указать em1 (вообще-то, указывать ничего не обязательно, можно выбрать «а» - автоопределение, уж чего определит – то и ладно, поскольку нам на первом этапе это не критично, правильное сопоставление нужно только после установки, но запомнить соответствие WAN=em0 (61-06), а LAN=em1 (6E-A7) – надо). Далее спросят про WAN – там (как ни странно) пишем em0.

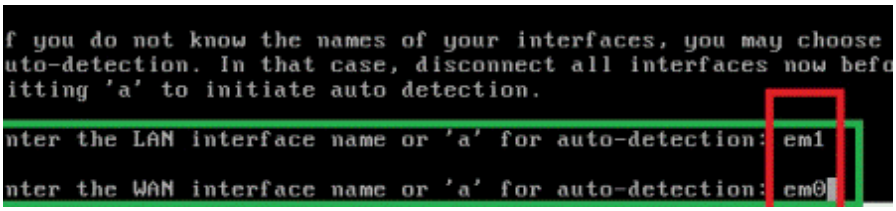


Рис.37 Настройка сетевых интерфейсов pfSence

а на вопрос о дополнительных интерфейсах не отвечать ничего (просто нажать ввод), тогда мы получим запрос подтверждения правильности сопоставления интерфейсов.

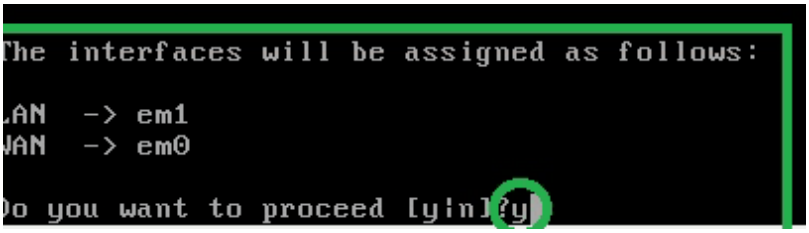


Рис.38 Запрос подтверждения правильности сопоставления интерфейсов

Теперь попадаем в меню, где выбираем 99 – установка на жесткий диск.

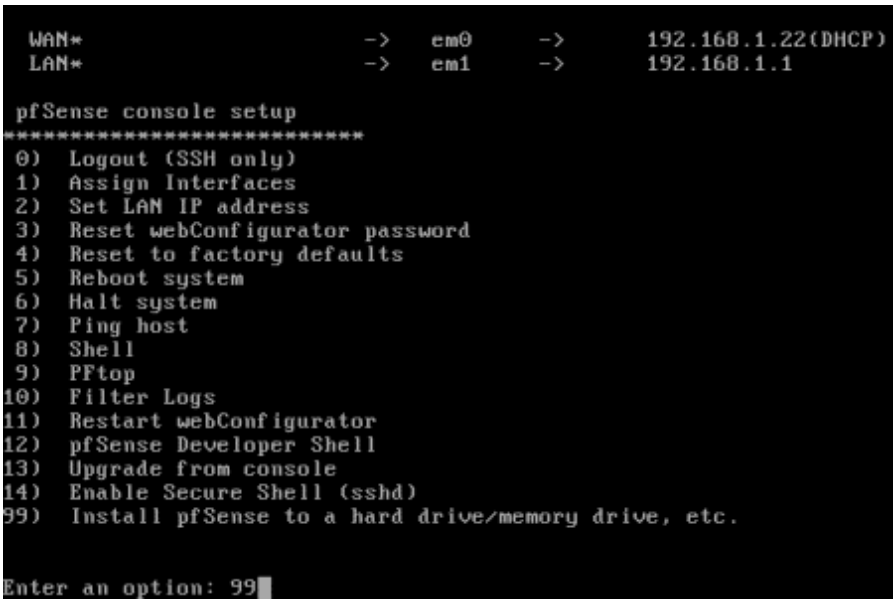


Рис.39 Выбор установки/работы дистрибутива

Тут же начинается установка, в первом окне спрашивается про локализацию шрифтов, я зачем-то выбрал так, как на картинке,

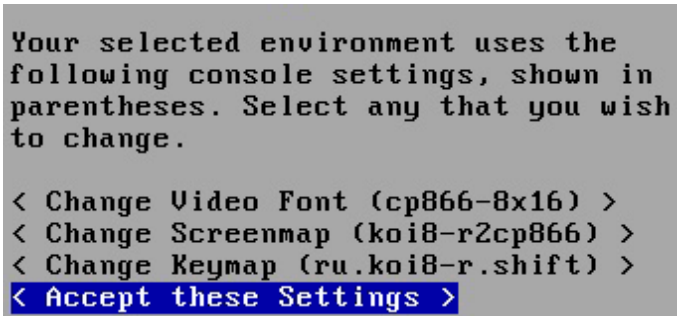


Рис.40 Настройка консоли

но подозреваю, что и при дефолтных установках результат будет тот же. Следующий экран – выбор способа установки.

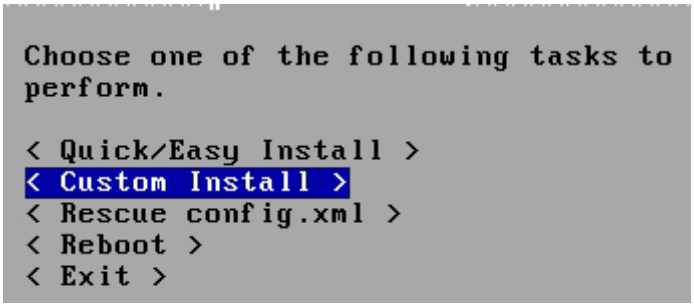


Рис.41 Выбор типа установки

На картинке выбран Custom (путь самурая). Нам достаточно “Quick/Easy Install”. Далее количество процессоров. Не буду объяснять почему – просто выбираем один.

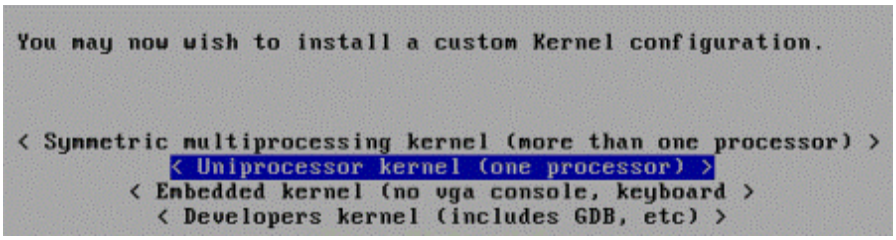


Рис.42 Настройка на использования процессоров

Операционная система установится, далее перезагрузите компьютер.

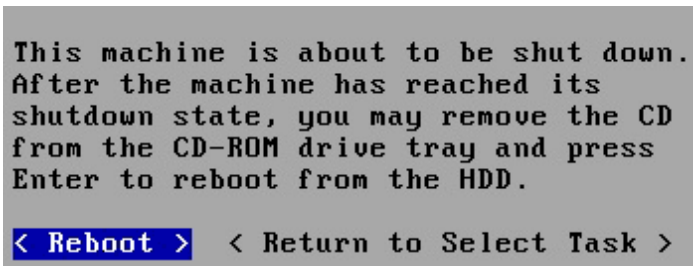


Рис.43, Диалог окончания установки

Если внимательно приглядеться, то там также написано: «Когда компьютер выключится, можно достать диск из CD привода. Я бы сказал, когда включится, но еще не начнет загружаться с диска. То есть во время тестирования BIOS надо изъять диск,

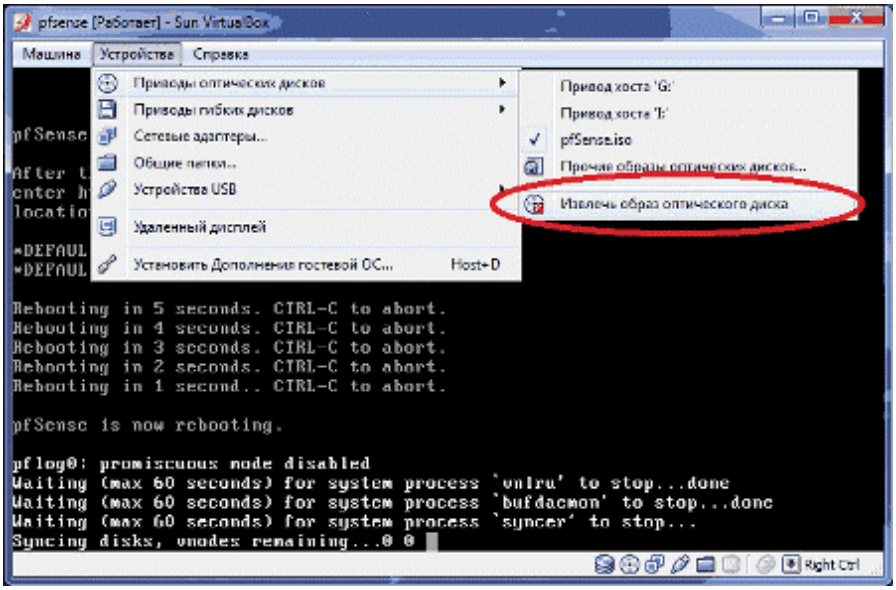


Рис.44 «Вытаскиваем» виртуальный CD-ROM из виртуальной машины

а то загрузка опять начнется с него. После загрузки получаем такое меню.

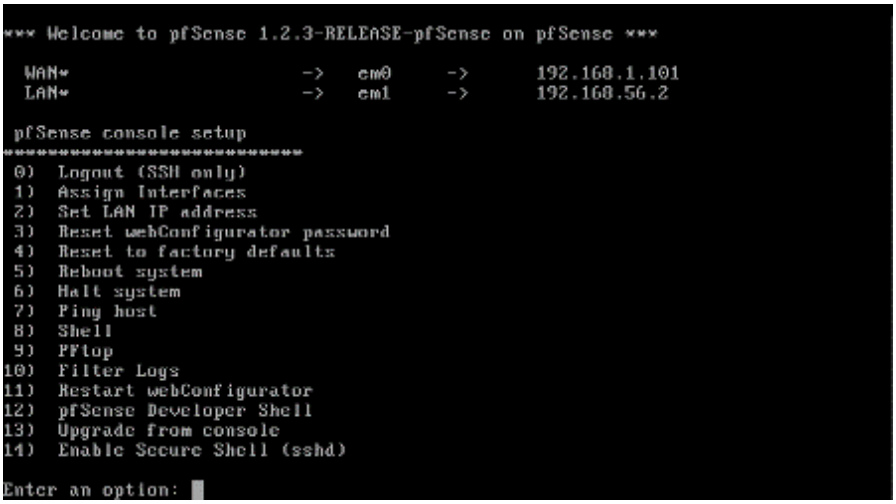


Рис.45 Меню запуска ОС

Если система неправильно определила соответствие LAN/WAN интерфейсов, тогда жмем 1) Assign Interfaces – назначить интерфейсы. Далее важно правильно указать адрес LAN для PFsense. Выбираем пункт 2 и далее, как на картинке.

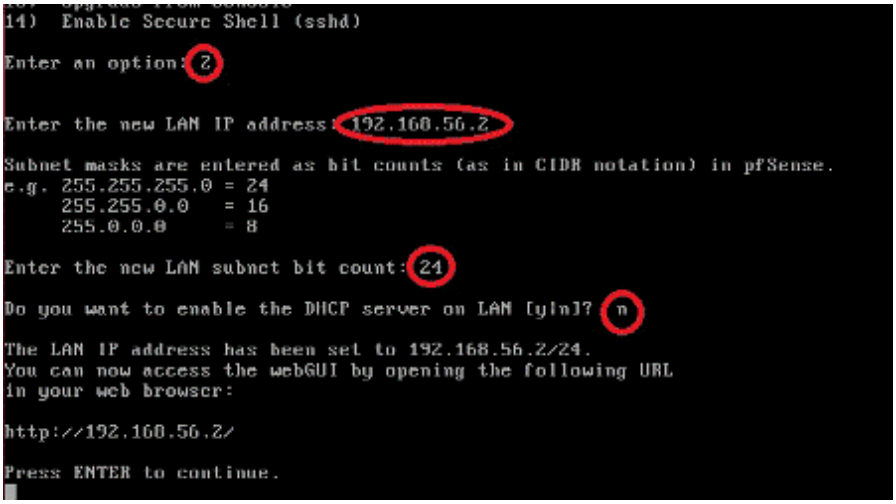


Рис.46 Проверяем правильность настроек

Почему адрес именно 192.168.56.2 – смотри выше. Он пишет, что теперь Вы можете попасть в консоль администрирования по адресу http://192.168.56.2 и просит нажать «Ввод». Нажимаем ввод и вбиваем в браузер этот адрес. Если все было сделано правильно, тогда мы попадаем на web интерфейс. Имя пользователя – admin, пароль – pfsense. После первого входа пароль надо, конечно, сменить. При первом входе мы попадаем в мастер общей настройки. Сначала ввод DNS серверов. Если внешний адрес динамический (получаем по DHCP) – указывать ничего не надо. Если статический – тогда указываем сервера (или как у меня – один сервер).

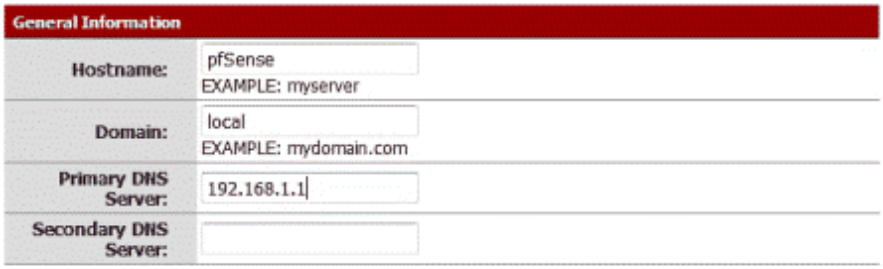


Рис.47 Основная информация

Следующий экран – настройка WAN.

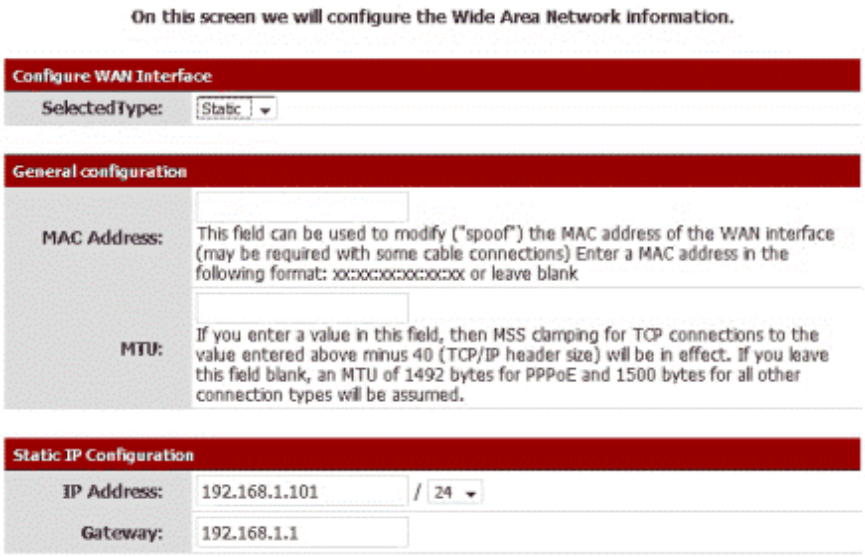


Рис.48 Конфигурирование настроек сети

Если WAN – динамический, то просто выбираем в чек-боксе dhcp, если статический, тогда выбираем static и указываем адрес, маску и шлюз. Если у Вас мудреный интернет, здесь же всю эту мудрость тоже можно отобразить, но это частности, отвлекаться не будем. Далее – Настройка LAN, мы его уже настроили, здесь трогать ничего не надо.

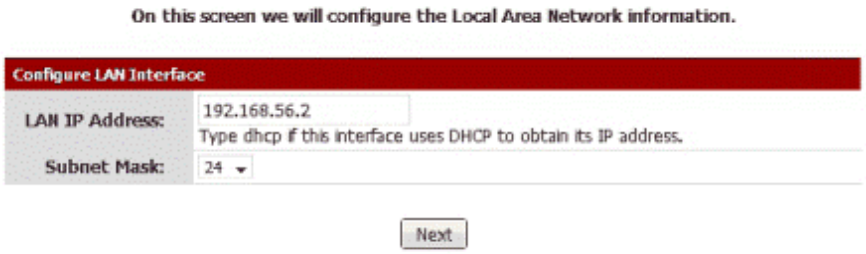


Рис.49 Настройка сети LAN

Последнее окно – смена пароля. Не мне Вам советовать, что делать. Если уж Вы дочитали до этого места, значит, поступите так, как надо. После этого Ваш браузер должен заиграть всеми красками интернета, но уже будучи загнанным в узкое русло сетевого экрана pf.

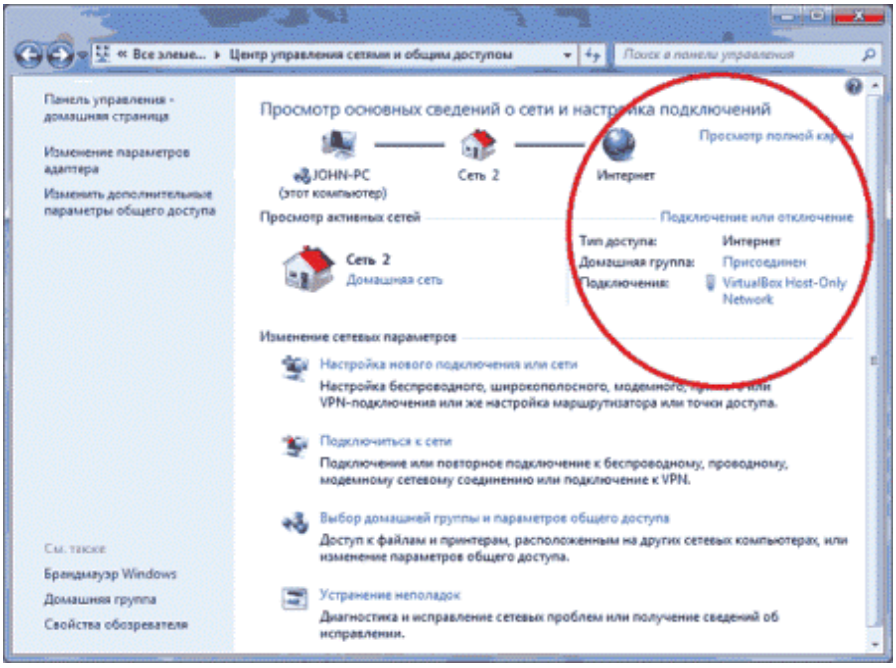


Рис.50 Проверка подключения к Интернет

По умолчанию входящие соединения запрещены, исходящие – разрешены. Такие настройки как раз подходят для простого пользователя, который опасается за свою безопасность, особенно при подключении с ноутбука в неизвестную локальную сеть (аэропорт, кафе, гостиница).

Но это еще не все. Чудесность VirtualBox заключается еще и в том, что в папке установки лежит утилита VBoxHeadless.exe, запуск которой с определенными параметрами, например, VBoxHeadless -s pfsense -v on -p 3000 -a 127.0.0.1 вызовет «безголовый» (безоконный) запуск виртуальной машины с названием pfsense, к которой при желании можно будет подключиться по RDP на адрес 127.0.0.1 (localhost) и порт 3000.

Создаем Bat-файл с этой строкой, кидаем его в автозапуск и через несколько минут после входа пользователя у Вас появляется интернет, проходящий через OpenBSD-ишный фаейрвол, портированный в FreeBSD, запущенной «втихую и втемную» на VirtualBoxe, захостенном на Windows 7.

Осознание самого этого факта преисполняет нормального параноика благоговением и душевным покоем. Дальнейшее совершенствование приветствуется. Настройка сетевого экрана как через веб-интерфейс, так и из консоли не знает границ.

Задание на лабораторную работу

1. Ознакомится с теоретической частью
2. В соответствии с рекомендация установить и настроить на виртуальной машине pfSence.
3. Исследовать работы фаервола:
    a. Использование Aliases
    b. Создание разрешающих/запрещающих правил, обеспечив фильтрацию трафика;
    c. Ограничение пропускной способности средствами фаервола;
    d. Создать правила работающие по времени;
    e. Проанализировать журнал работы фаервола
4. Исследовать дополнительные функции pfSence:
    a. Выбрать две понравившиеся дополнительные функции и реализовать их работу;
    b. Проанализировать процесс работы двух выбранных дополнительных функций.
5. Подготовить отчет о проделанной работе.

Требования к содержанию отчета

Отчёт должен содержать:

6. Краткие теоретические сведения.
7. Задание.
8. Результаты выполнения задания.
9. Результаты выполнения программного диалога.
10. Выводы по работе.

Литература

14. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.

15. Одом, Уэнделл. Официальное руководство Cisco по подготовке к сертифицированным экзаменам CCENT/CCNA ICND1. 2-е изд. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011 – 672с.

16. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.

17. http://www.prevelakis.net/Papers/VirtualFirewall.pdf «The Virtual Firewall» Vassilis Prevelakis Computer Science Department Drexel University

18. https://www.pfsense.org/get-support/index.html#documentation //Документация по pfSance

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Дальневосточный федеральный университет»

ШКОЛА ЕСТЕСТВЕННЫХ НАУК
Кафедра информационных систем управления

С. С. Пашин

ЭМУЛЯТОР РАБОТЫ СЕТИ CISCO PACKET TRACER

Методические указания к лабораторной работе
по дисциплине «Сети ЭВМ и телекоммуникации»

Владивосток
2013

Приведены основные сведения построения эмуляционных моделей работы сети передачи данных на примере оборудования фирмы CISCO. Цель указаний – формирование у студентов навыков построения сетей передачи данных, а так же рассмотрение функциональных особенностей основных элементов компьютерной сети, таких как маршрутизаторы, свичи и хабы.

Рассчитаны на студентов по направлению подготовки 230100.62 – “Информатика и вычислительная техника” для профиля подготовки “Автоматизированные системы обработки информации и управления” (квалификация «бакалавр»), а также для студентов других специальностей, изучающих методы обработки и передачи информации

Составитель ассистент кафедры ИСУ Пашин С.С.

Содержание

Введение

Разные виды эмуляции широко используются при разработке и проектировании новых систем. Эмуляция упрощает разработку, давая возможность определить, исследовать и устранить недостатки проекта до его физического воплощения. Особенно эмуляция полезна при разработке дорогих и сложных систем, в которых конфликты параллельной обработки часто достаточно сложно определить и диагностировать без применения виртуальной управляемой аппаратуры, доступной при эмуляции. Также эмуляция позволяет приступить к разработке сложных систем до фактической покупке или изготовления аппаратной части, таким образом, проверяя заложенные в нее особенности.

Cisco Packet Tracer — эмулятор сети передачи данных, выпускаемый фирмой Cisco Systems. Позволяет делать работоспособные модели сети, настраивать маршрутизаторы и коммутаторы, взаимодействовать между несколькими пользователями. Включает в себя серии маршрутизаторов Cisco 1800, 2600, 2800 и коммутаторов 2950, 2960, 3650. Кроме того есть серверы DHCP, HTTP, TFTP, FTP, рабочие станции, различные модули к компьютерам и маршрутизаторам, устройства WiFi, различные кабели.

Первым этапом проектирования компьютерной сети является выбор технических средств и системы протоколов (включая способы коммутации и доставки данных в региональной и базовой сети). Второй этап проектирования требует решения совокупности сложных взаимосвязанных задач, к которым относятся: оптимизация пропускной способности каналов связи; выбор маршрутов; оптимизация топологической структуры; выбор методов управления потоками и определение параметров управления; анализ объемов буферной памяти узлов коммутации и маршрутизации и выбор стратегии буферизации при перегрузках и т.д.

На сегодняшний день на рынке IT существуют не так уж и много сетевых симуляторов.

Широко известны такие симуляторы, как:

- BOSON NET SIM;
- CISCO Router eSim;
- Cisco Packet Tracker;
- Network Emulator;
- Dynamips;
- Cisco 7200 Simulator.

Из них наиболее распространенные в плане использования для обучения являются Boson NetSim, Cisco Packet Tracer и Network Emulator. Остановимся на них подробней.

1. Обзор средств эмуляции

1.1 Boson NetSim

Boson NetSim – программное обеспечение, которое моделирует работу сетевого оборудования Cisco, и разработано, чтобы помочь пользователю в изучении Cisco IOS.

Большинство других программных продуктов, «моделируя» поведение системы в заранее подготовленных лабораторных работах, фактически не могут отображать ситуаций, которые действительно могут случиться в сети. В отличие от них, NetSim использует технологии, специально разработанные компанией Boson, которые позволяют обойти этот недостаток и моделировать истинное поведение сети. Эти технологии позволят многим пользователям Boson NetSim выйти далеко за рамки выдуманных лабораторных работ, и лучше понять принципы функционирования Cisco IOS.

NetSim имеет очень развитую поддержку, обеспечивающую компанией Boson (это связано, конечно же, с бурными темпами развития телекоммуникационных сетей). В связи с этим, компания Cicso рекомендует использовать этот продукт для подготовки к сдачи экзаменов. Поэтому Boson выпускает различные версии NetSim’а, каждая из которых ориентирована на определенный экзамен и, соответственно, уровень знания пользователя.

Существует три версии NetSim’a для следующих экзаменов: NetSim для CCENT, NetSim для CCNA и NetSim для CCNP.

1.2 Cisco Packet Tracer

Данный программный продукт разработан компанией Cisco и рекомендован использоваться при изучении телекоммуникационных сетей и сетевого оборудования.

Packet Tracer 4.0 включает следующие особенности:

- моделирование логической топологии: рабочее пространство для того, чтобы создать сети любого размера на CCNA-уровне сложности;
- моделирование в режиме реального времени;
- режим симуляции;
- моделирование физической топологии: более понятное взаимодействие с физическими устройствами, используя такие понятия как город, здание, стойка и т.д.;
- улучшенный GUI, необходимый для более качественного понимания организации сети, принципов работы устройства;
- многоязыковая поддержка: возможность перевода данного программного продукта практически на любой язык, необходимый пользователю;
- усовершенствованное изображение сетевого оборудования со способностью добавлять / удалять различные компоненты;
- наличие Activity Wizard позволяет студентам и преподавателям создавать шаблоны сетей и использовать их в дальнейшем.

С помощью данного программного продукта преподаватели и студенты могут придумывать, строить, конфигурировать сети и производить в них поиск неисправностей. Packet Tracer дает возможность более подробно представлять новейшие технологии, тем самым делая учебный процесс чрезвычайно полезным с точки зрения усвоения полученного материала.

1.3 Network Emulator

Программа Network Emulator была задумана в начале 1997 года. Проект превратился, по сути, в программу, обучающую ее пользователя всем тонкостям технологии на разных уровнях: от базовых понятий до особенностей обработки отдельных полей сетевых пакетов. Программа прошла путь от простейшего «роутера пакетов» до интеллектуального организатора виртуальных машин: на любом из компьютеров можно запустить несколько программ-аналогов настоящих приложений. Все они будут исполняться одновременно.

В дальнейшем появилось и другое «призвание» Network Emulator: обучение студентов принципу администрирования IP-сетей. Данное направление использования было с успехом реализовано в процессе проведения лабораторных работ по предмету "Сети ЭВМ" в Ульяновском Государственном Техническом Университете.

Данный симулятор включает в себя следующие возможности и технологии:

- маршрутизация, система моделирования каналов, IP-фильтрация;
- типы пакетов: ICMP, UDP, TCP, а так же низкоуровневые ARP-запросы;
- концепция интерфейсов и сокетов (простой, дейтаграммный и потоковый);
- эмуляция хостов, коммутаторов второго уровня и концентраторов;
- установка уровня помех на канале;
- связывание нескольких Network Emulator через реальную сеть TCP/IP.

В рамках курса лабораторные работы будут выполняться на Cisco Packet Tracer.

Cisco Packet Tracer специально разработан для начала изучения современных телекоммуникационных систем, и больше других симуляторов соответствует данной задаче.

2 Cisco Packet Tracer

Данный симулятор позволяет студентам проектировать свои собственные сети, создавая и отправляя различные пакеты данных, сохранять и комментировать свою работу. Студенты могут изучать и использовать такие сетевые устройства, как коммутаторы второго и третьего уровней, рабочие станции, определять типы связей между ними и соединять их. После того, как сеть спроектирована, студенты могут приступать к конфигурированию выбранных устройств посредством терминального доступа или командной строки (см. рис.2.1).

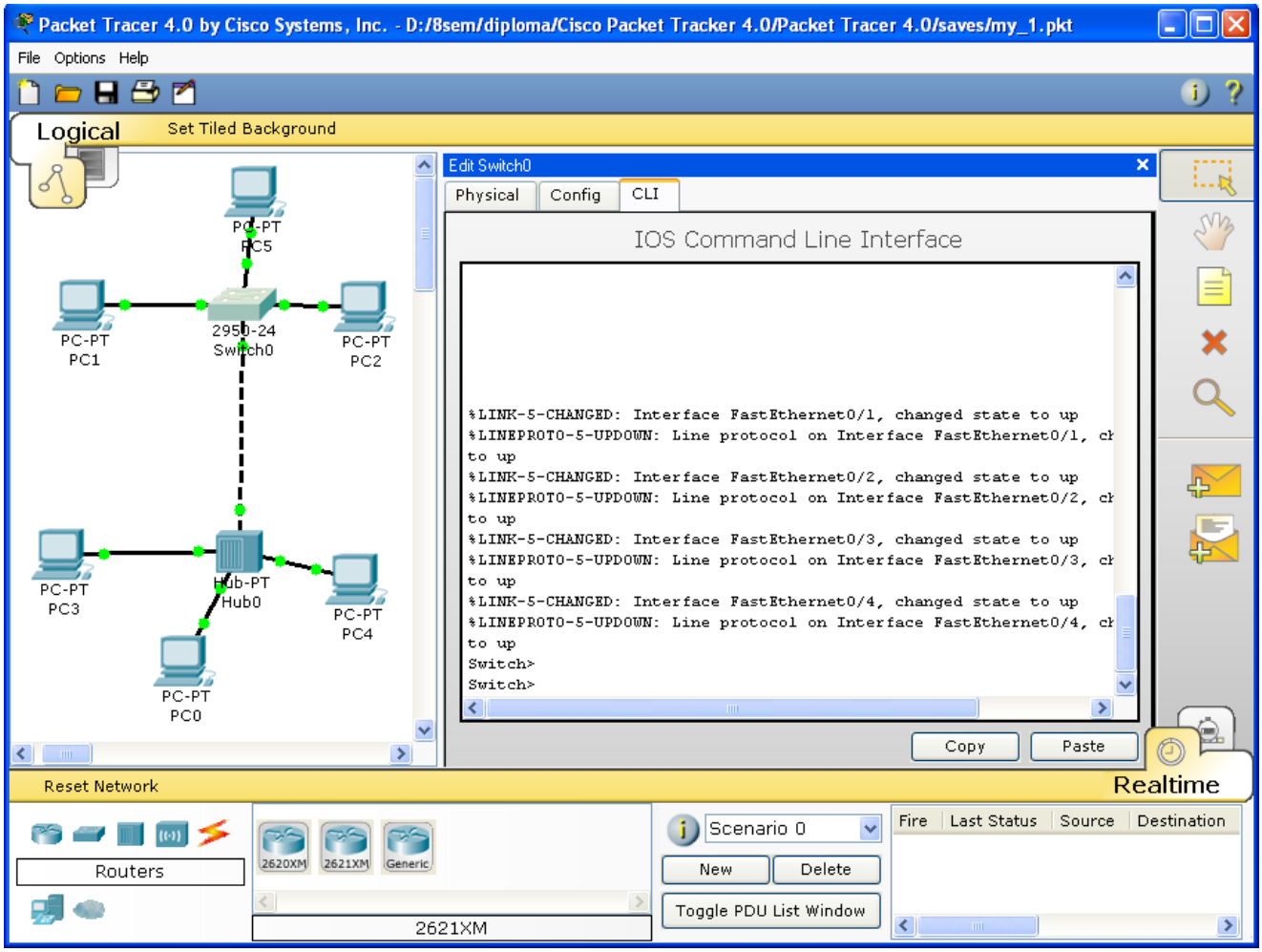


Рис.2.1 Cisco Packet Tracer 4.0

Отличительной особенностью данного симулятора является наличие в нем «Режима симуляции» (рис.2.2). В данном режиме все пакеты, пересылаемые внутри сети, отображаются графически. Эта возможность позволяет студентам наглядно продемонстрировать, по какому интерфейсу в данные момент перемещается пакет, какой протокол используется и т.д.

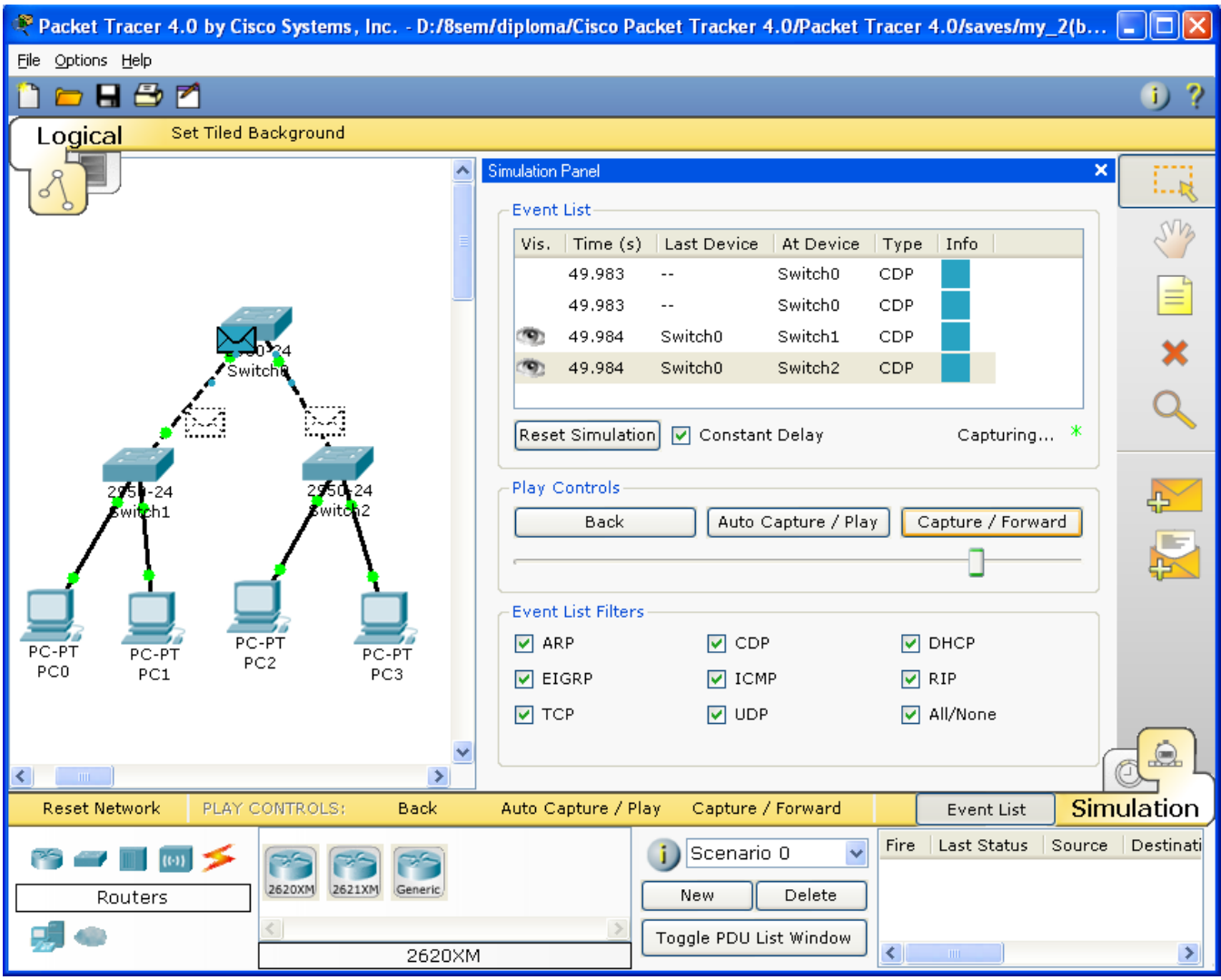


Рис.2.2 Режим «Симуляции» в Cisco Packet Tracer 4.0

Однако, это не все преимущества Packet Tracer: в «Режиме симуляции» студент может не только отслеживать используемые протоколы, но и видеть, на каком из семи уровней модели OSI данный протокол задействован (см.рис.2.3).

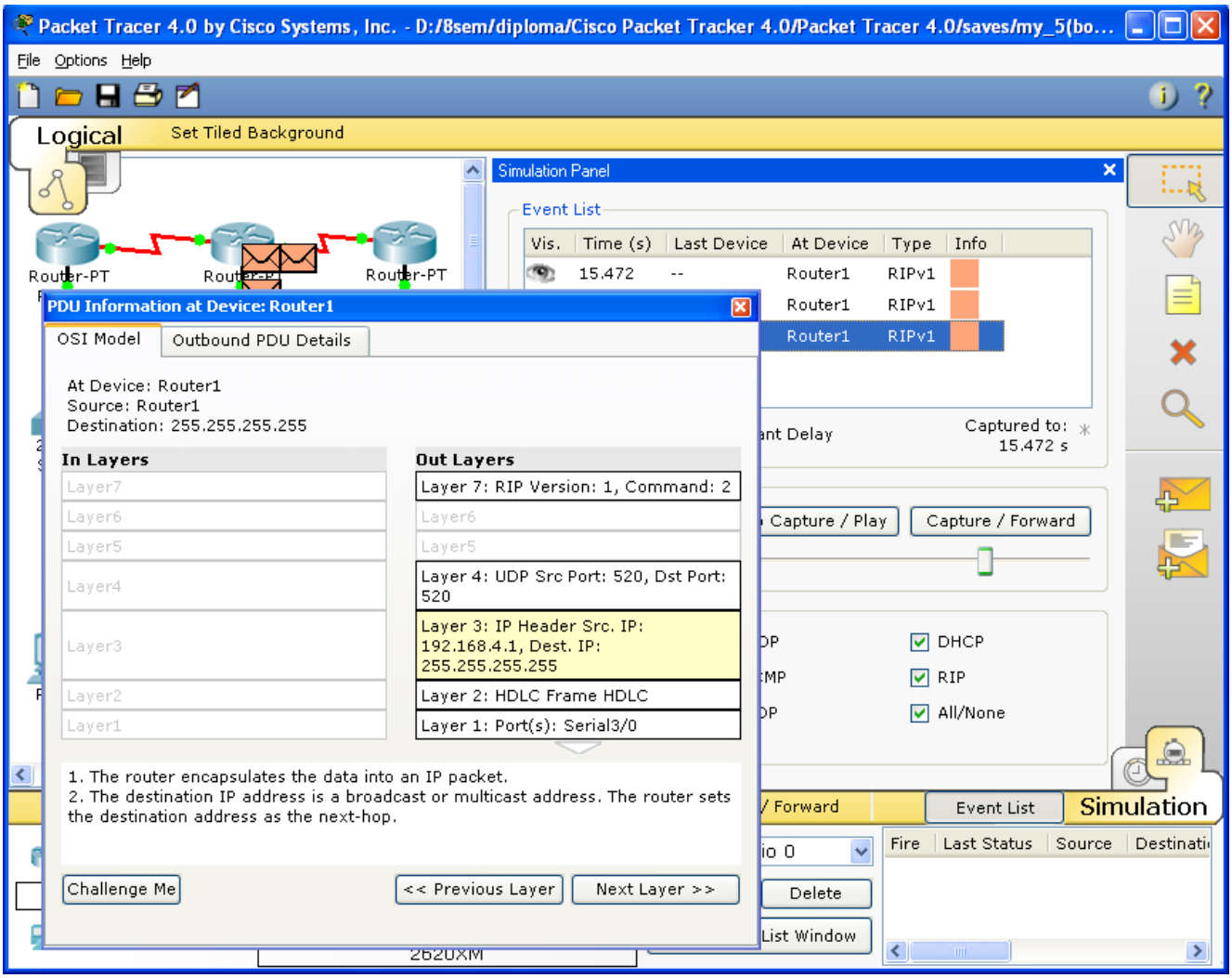


Рис.2.3 Анализ семиуровневой модели OSI в Cisco Packet Tracer 4.0

Такая кажущаяся на первый взгляд простота и наглядность делает практические занятия чрезвычайно полезными, совмещая в них как получение, так и закрепление полученного материала.

Packet Tracer способен моделировать большое количество устройств различного назначения, а так же немало различных типов связей, что позволяет проектировать сети любого размера на высоком уровне сложности:

моделируемые устройства:

- коммутаторы третьего уровня:
  - Router 2620 XM;
  - Router 2621 XM;
  - Router-PT.

- Коммутаторы второго уровня:
    - Switch 2950-24;
    - Switch 2950T;
    - Switch-PT;
    - соединение типа «мост» Bridge-PT.
- Сетевые концентраторы:
    - Hub-PT;
    - повторитель Repeater-PT.
- Оконечные устройства:
    - рабочая станция PC-PT;
    - сервер Server-PT;
    - принтер Printer-PT.
- Беспроводные устройства:
    - точка доступа AccessPoint-PT.
- Глобальная сеть WAN.

Типы связей:

- консоль;
- медный кабель без перекрещивания (прямой кабель);
- медный кабель с перекрещиванием (кросс-кабель);
- волоконно-оптический кабель;
- телефонная линия;
- Serial DCE;
- Serial DTE.

Так же целесообразно привести те протоколы, которые студент может отслеживать:

- ARP;
- CDP;
- DHCP;
- EIGRP;
- ICMP;

- RIP;
- TCP;
- UDP.

3 Описание терминального режима

Маршрутизатор конфигурируется в командной строке операционной системы Cisco IOS. Подсоединение к маршрутизатору осуществляется через Telnet на IP-адрес любого из его интерфейсов или с помощью любой терминальной программы через последовательный порт компьютера, связанный с консольным портом маршрутизатора. Последний способ предпочтительнее, потому что процесс конфигурирования маршрутизатора может изменять параметры IP-интерфейсов, что приведет к потере соединения, установленного через Telnet. Кроме того, по соображениям безопасности доступ к маршрутизатору через Telnet следует запретить.

В рамках данного курса конфигурация маршрутизаторов будет осуществляться посредством терминала.

При работе в командной строке Cisco IOS существует несколько контекстов (режимов ввода команд).

Контекст пользователя открывается при подсоединении к маршрутизатору; обычно при подключении через сеть требуется пароль, а при подключении через консольный порт пароль не нужен. В этот же контекст командная строка автоматически переходит при продолжительном отсутствии ввода в контексте администратора. В контексте пользователя доступны только простые команды (некоторые базовые операции для мониторинга), не влияющие на конфигурацию маршрутизатора. Вид приглашения командной строки:

```
router>
```

Вместо слова router выводится имя маршрутизатора, если оно установлено.

Контекст администратора (контекст "exec") открывается командой **enable**, поданной в контексте пользователя; при этом обычно требуется пароль администратора. В контексте администратора доступны команды, позволяющие получить полную информацию о конфигурации маршрутизатора и его состоянии, команды перехода в режим конфигурирования, команды сохранения и загрузки конфигурации. Вид приглашения командной строки:

router#

Обратный переход в контекст пользователя производится по команде **disable** или по истечении установленного времени неактивности. Завершение сеанса работы - команда **exit.**

Глобальный контекст конфигурирования открывается командой **config terminal** ("конфигурировать через терминал"), поданной в контексте администратора. Глобальный контекст конфигурирования содержит как непосредственно команды конфигурирования маршрутизатора, так и команды перехода в контексты конфигурирования подсистем маршрутизатора, например:

Router(config)#

контекст конфигурирования интерфейса

открывается командой **interface *имя_интерфейса*** (например **interface serial0**), поданной в глобальном контексте конфигурирования;

контекст конфигурирования процесса динамической маршрутизации

открывается командой **router** ***протокол номер_процесса*** (например, **router ospf 1**, поданной в глобальном контексте конфигурирования.

Существует множество других контекстов конфигурирования. Некоторые контексты конфигурирования находятся внтури других контекстов конфигурирования.

Вид приглашения командной строки в контекстах конфигурирования, которые будут всречаться наиболее часто:

router(config)# /глобальный/
router(config-if)# /интерфейса/
rounter(config-router)# /динамической маршрутизации/
rounter(config-line)# /терминальной линии/

ВАЖНО! *Студенты должны запомнить вид приглашений командой строки во всех вышеуказанных контекстах и правила перехода из контекста в контекст. В дальнейшем примеры команд всегда будут даваться вместе с приглашениями, из которых студенты должны определять контекст, в котором подается команда. Примеры не будут содержать указаний, как попасть в необходимый контекст.*

Выход из глобального контекста конфигурирования в контекст администратора, а также выход из любого подконтекста конфигурирования в контекст верхнего уровня производится командой **exit** или **Ctrl-Z**. Кроме того, команда **end**, поданная в любом из контекстов конфигурирования немедленно завершает процесс конфигурирования и возвращает оператора в контекст администратора.

ВАЖНО! *Любая команда конфигурации вступает в действие немедленно после ввода, а не после возврата в контекст администратора.*

Упрощенная схема контекстов представлена на рис.3.1.

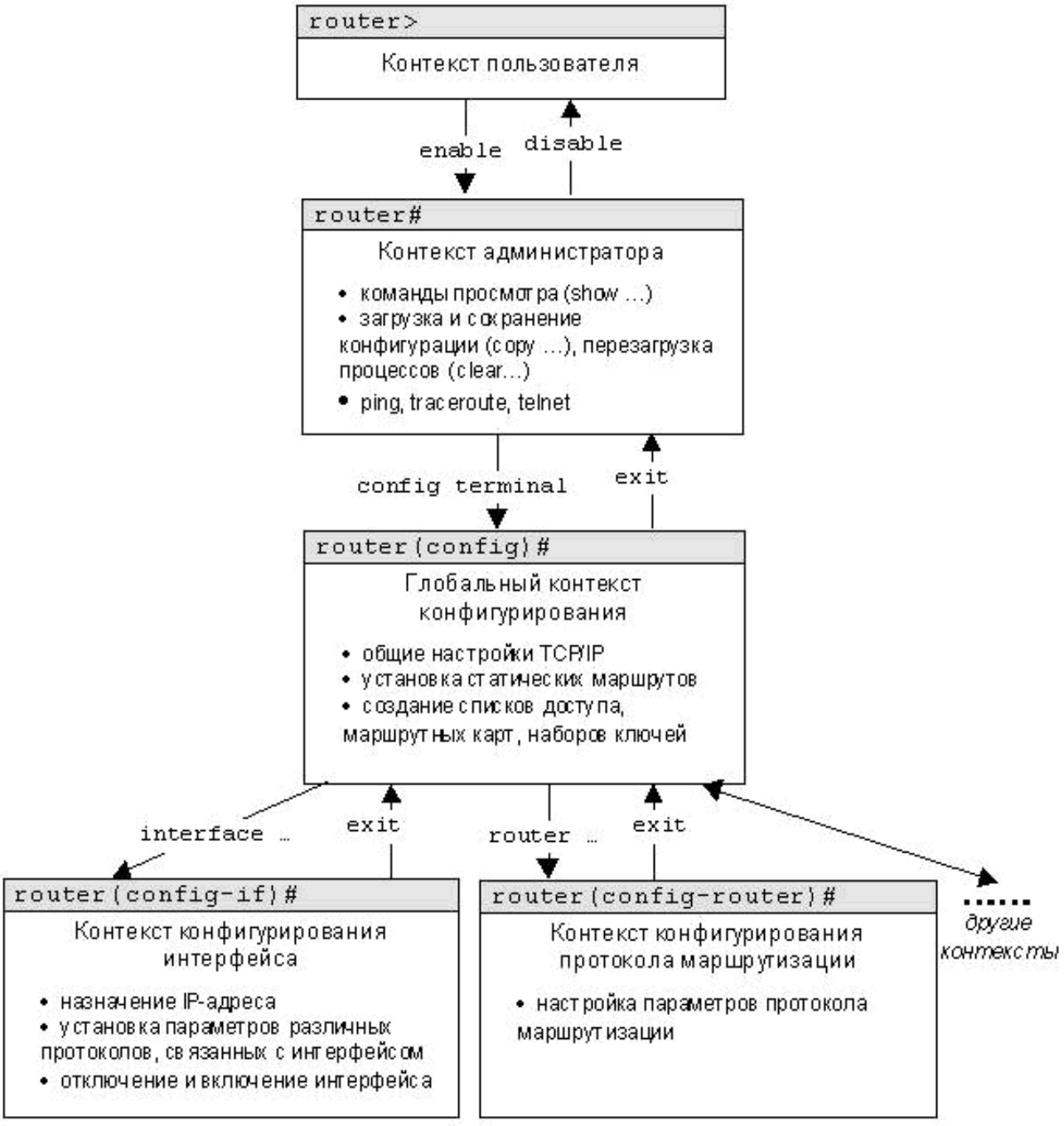


Рис.3.1. Схема контекстов Cisco IOS

Все команды и параметры могут быть сокращены (например, "**enable**" - "**en**", "**configure terminal**" - "**conf t**"); если сокращение окажется неоднозначным, маршрутизатор сообщит об этом, а по нажатию табуляции выдаст варианты, соответствующие введенному фрагменту.

В любом месте командной строки для получения помощи может быть использован вопросительный знак:

router#**?** /список всех команд данного контекста с комментариями/

router#**co?** /список всех слов в этом контексте ввода, начинающихся на "co" - нет пробела перед "?"/

router#**conf ?** /список всех параметров, которые могут следовать за командой config - перед "?" есть пробел/

4 Список команд

Данный список команд сгруппирован в соответствии с контекстами, в котором они [команды] применяются. В данном списке собраны те команды конфигурирования, которые необходимы для выполнения всех лабораторных работ.

4.1 Глобальный контекст конфигурирования

4.1.1 Команда «Access-list»

Критерии фильтрации задаются в списке операторов разрешения и запрета, называемом списком доступа. Строки списка доступа сравниваются с IP-адресами и другой информацией пакета данных последовательно в том порядке, в котором были заданы, пока не будет найдено совпадение. При совпадении осуществляется выход из списка. При этом работа списка доступа напрямую зависит от порядка следования строк.

Списки доступа имеют 2 *правила*: permit – разрешить, и deny – запретить. Именно они определяют, пропустить пакет дальше или запретить ему доступ.

Списки доступа бывают 2-ух типов: standard – стандартные (номера с 1 до 99) и extended – расширенные (номера с 100 до 199). Различия заключаются в возможности фильтровать пакеты не только по ip-адресу, но и по другим параметрам.

Формат команды (стандартные списки доступа):

access-list *номер_списка/имя правило A.B.C.D a.b.c.d* , где A.B.C.D a.b.c.d – ip-адрес и подстановочная маска соответственно.

Пример выполнения команды:

```
Router(config)#access-list 10 deny 192.168.3.0 0.0.0.3
Router(config)#
```

Данная команда означает, что данный список доступа блокирует любые пакеты с ip-адресами 192.168.3.1 - 192.168.3.3.

4.1.2 Команда «Enable secret»

Обычно при входе в привилегированный режим требуется ввести пароль. Данная функция позволяет предотвратить несанкционированный доступ в данный режим, ведь именно из него можно изменять конфигурацию устройства. Данная команда позволяет установить такой пароль.

Формат команды:

enable secret *пароль*

Пример выполнения команды:

```
Switch(config)#enable secret 123
Switch(config)#
%SYS-5-CONFIG_I: Configured from console by console
Switch#exit
Switch con0 is now available
Press RETURN to get started.
```

```
Switch>enable
Password:
Switch#
```

После того, как был установлен пароль, при попытке входа в привилегированный режим, коммутатор будет требовать от пользователя его ввести – в противном случае вход будет невозможен.

4.1.3 Команда «Interface»

Команда для входа в режим конфигурирования интерфейсов конфигурируемого устройства. Данный режим представляет собой одно из подмножеств режима глобального конфигурирования и позволяет настраивать один из доступных сетевых интерфейсов (fa 0/0, s 2/0 и т.д.). Все изменения, вносимые в конфигурацию коммутатора в данном режиме относятся только к выбранному интерфейсу.

Формат команды (возможны 3 варианта):

interface *тип порт*

interface *тип слот/порт*

interface *тип слот/подслот/порт*

```
Switch(config)#interface vlan 1
Switch(config-if)#

Router(config)#interface s 3/0
Router(config-if)#
```

Примеры выполнения команды:

После введения данной команды с указанным интерфейсом пользователь имеет возможность приступить к его конфигурированию. Необходимо

заметить, что, находясь в режиме конфигурирования интерфейса, вид приглашения командной строки не отображает имя данного интерфейса.

4.1.4 Команда «Ip route»

Статическая маршрутизация предполагает фиксированную структуру сети: каждый маршрутизатор в сети точно знает, куда нужно отправлять пакет, чтобы он был доставлен по назначению. Для этого можно прописать статические маршруты, используя данную команду. Команда может быть записана в двух форматах:

Первый формат команды:

ip route *A.B.C.D a.b.c.d A1.B1.C1.D1* ,

где A.B.C.D и a.b.c.d – сетевой адрес и маска подсети, куда необходимо доставить пакеты, A1.B1.C1.D1 – ip-адрес следующего маршрутизатора в пути или адрес сети другого маршрутизатора из таблицы маршрутизации, куда должны переадресовываться пакеты;

Второй формат команды:

ip route *A.B.C.D a.b.c.d выходной_интерфейс_текущего_маршрутизатора*

```
Router(config)#ip route 76.115.253.0 255.0.0.0 76.115.252.0
Router(config)#
Router(config)#ip route 0.0.0.0 0.0.0.0 Serial2/0
Router(config)#
```

Примеры выполнения команды:

Данной командой указывается маршрут, по которому пакеты из одной подсети будут доставляться в другую. Маршрут по умолчанию (Router(config)#ip route 0.0.0.0 0.0.0.0 serial 2/0) указывает, что пакеты, предназначенные узлам в другой подсети должны отправляться через данный шлюз.

4.1.5 Команда «Hostname»

Данная команда используется для изменения имени конфигурируемого устройства.

Формат команды:

hostname *новое_имя*

Пример выполнения команды:

```
Router(config)#hostname R1
R1(config)#
```

Как видно, маршрутизатор поменял своё имя с Router на R1.

4.1.6 Команда «Router rip»

RIP – Routing Information Protocol – протокол динамической маршрутизации. При его использовании отпадает необходимость вручную прописывать все маршруты – необходимо лишь указать адреса сетей, с которыми нужно обмениваться данными. Данная команда позволяет включить rip-протокол.

Пример выполнения команды:

```
Router(config)#router rip
Router(config-router)#
```

Данная команда включает rip-протокол на данном маршрутизаторе. Дальнейшая настройка производится из соответствующего контекста маршрутизации, описанного отдельно.

4.2 Контекст конфигурирования интерфейса

4.2.1 Команда «Ip access-group»

Данная команда используется для наложения списков доступа. Список накладывается на конкретный интерфейс, и указывается один из 2-ух параметров: in (на входящие пакеты) или out (на исходящие). Необходимо знать, что на каждом интерфейсе может быть включен только один список доступа.

Формат команды:

ip access-group *номер_списка/имя_параметр*

Пример выполнения команды:

```
Router(config-if)# ip access group 10 in
Router(config-if)#
```

В данном примере да выбранный интерфейс накладывается список доступа под номером 10: он будет проверять все входящие в интерфейс пакеты, так как выбран параметр in.

4.2.2 Команда «Bandwidth»

Данная команда используется только в последовательных интерфейсах и служит для установки ширины полосы пропускания. Значение устанавливается в килобитах.

Формат команды:

bandwidth *ширина_полосы_пропускания*

Пример выполнения команды:

```
Router(config)#interface serial 2/0
Router(config-if)#bandwidth 560
```

```
Router(config-if)#
```

После выполнения данной команды ширина полосы пропускания для serial 2/0 будет равна 560 kbits.

4.2.3 Команда «Clock rate»

Для корректной работы участка сети, где используется последовательный сетевой интерфейс, один из коммутаторов 3-его уровня должен предоставлять тактовую частоту. Это может быть оконечное кабельное устройство DCE (расшифровать). Так как маршрутизаторы CISCO являются по умолчанию устройствами DTE, то необходимо явно указать интерфейсу на предоставление тактовой частоты, если этот интерфейс работает в режиме DCE. Для этого используют данную команду (значение устанавливается в битах в секунду).

Формат команды:

clock rate *тактовая_частота*

Пример выполнения команды:

```
Router(config)#interface serial 2/0
Router(config-if)#clock rate 56000
Router(config-if)#
```

После выполнения данной команды тактовая частота для serial 2/0 будет равна 56000 bits per second.

4.2.4 Команда «Ip address»

Каждый интерфейс должен обладать своим уникальным ip-адресом – иначе взаимодействие устройств по данному интерфейсу не сможет быть

осуществлено. Данная команда используется для задания ip-адреса выбранному интерфейсу.

Формат команды:

ip address *A.B.C.D a.b.c.d* ,

где A.B.C.D a.b.c.d – ip-адрес и маска подсети соответственно.

Пример выполнения команды:

```
Switch(config)#interface vlan 1
Switch(config-if)#ip address 172.16.10.5 255.255.0.0
Switch(config-if)#
```

Результат можно проверить командой

Switch#show ip interface vlan 1

Данной командой интерфейсу vlan 1 назначен ip-адрес 172.16.10.5 с маской подсети 255.255.0.0.

4.2.5 Команда «No»

Данная команда применяется в случае необходимости отменить действие какой-либо команды конфигурирования.

Формат команды:

no *команда_которую_следует_отменить*

Пример выполнения команды:

```
Switch(config-if)# no shutdown
%LINK-5-CHANGED: Interface Vlan1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to up
Switch(config-if)#
```

В данном примере использовалась команда shutdown, которая отключает выбранный интерфейс. В итоге после выполнения no shutdown интерфейс включается.

4.3 Контекст администратора

4.3.1 Команда «Configure terminal»

Для конфигурирования устройства, работающего под управлением IOS, следует использовать привилегированную команду configure. Эта команда переводит контекст пользователя в так называемый «режим глобальной конфигурации» и имеет три варианта:

- конфигурирование с терминала;
- конфигурирование из памяти;
- конфигурирование через сеть.

В рамках данного лабораторного курса конфигурирование будет производиться **только** посредством терминала.

Из режима глобальной конфигурации можно делать изменения, который касаются устройства в целом. Также данный режим позволяет входить в режим конфигурирования определенного интерфейса.

Пример выполнения команды:

```
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

Переход в режим глобальной конфигурации, о чем свидетельствует изменившийся вид приглашения командной строки.

4.3.2 Команда «Copy»

После настройки коммутатора рекомендуется сохранять его текущую конфигурацию. Информация помещается в энергонезависимую память и

хранится там столько, сколько нужно. При необходимости все настройки могу быть восстановлены или сброшены.

Формат команды:

copy *running-config startup-config* – команда для сохранения конфигурации

copy *startup-config running-config* – команда для загрузки конфигурации

Пример выполнения команды:

```
Switch#copy running-config startup-config
Building configuration...
[OK]
Switch#
```

В данном примере текущая конфигурация коммутатора была сохранена в энергонезависимую память.

4.3.3 Команда «Show»

Show (англ. - показывать) – одна из наиболее важных команд, использующихся при настройке коммутаторов. Она применяется для просмотра информации любого рода и применяется практически во всех контекстах. Эта команда имеет больше всех параметров.

Здесь будут рассмотрены только те параметры, которые требуются в рамках данного курса. Другие параметры студент может изучить самостоятельно.

4.3.3.1 Параметр «running-config» команды «Show»

Для просмотра текущей работающей конфигурации коммутатора используется данная команда.

Пример выполнения команды:

```
Switch#show running-config
!
version 12.1
!
hostname Switch
...
```

На экран выводится текущие настройки коммутатора.

4.3.3.2 Параметр «startup-config» команды «Show»

Для просмотра сохраненной конфигурации используется данная команда.

Пример выполнения команды:

```
Switch#show startup-config
Using 1540 bytes
!
version 12.1
!
...
```

Если энергонезависим память не содержит информации, тогда коммутатор выдаст сообщение о том, что конфигурация не была сохранена.

Пример выполнения команды:

```
Switch #show startup-config
startup-config is not present
Switch #
```

Вывод сообщения о том, что в памяти отсутствует какая-либо информация.

4.3.3.3 Параметр «ip route» команды «Show»

Данная команда применяется для просмотра таблицы маршрутов.

Пример выполнения команды:

```
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
Gateway of last resort is 0.0.0.0 to network 0.0.0.0
C    192.168.1.0/24 is directly connected, FastEthernet0/0
C    192.168.2.0/24 is directly connected, Serial2/0
S    192.168.3.0/24 is directly connected, Serial2/0
S    192.168.4.0/24 is directly connected, Serial2/0
S    192.168.5.0/24 is directly connected, Serial2/0
S*   0.0.0.0/0 is directly connected, Serial2/0
Router#
```

Производится вывод таблицы маршрутизации.

4.3.3.4 Параметр «ip protocols» команды «Show»

Данная команда используется для просмотра протоколов маршрутизации, включенных на данном устройстве.

Пример выполнения команды:

```
Router#show ip protocols
```

```
Routing Protocol is "rip"
Sending updates every 30 seconds, next due in 18 seconds
Invalid after 180 seconds, hold down 180, flushed after 240
Outgoing update filter list for all interfaces is not set
Incoming update filter list for all interfaces is not set
Redistributing: rip
Default version control: send version 1, receive any version
  Interface          Send  Recv  Triggered RIP  Key-chain
  FastEthernet0/0    1     2 1
  Serial2/0          1     2 1
Automatic network summarization is in effect
Maximum path: 4
Routing for Networks:
     192.168.1.0
     192.168.2.0
Passive Interface(s):
Routing Information Sources:
     Gateway         Distance      Last Update
     192.168.2.2          120
Distance: (default is 120)
Router#
```

Выводится информация о включенных протоколах маршрутизации.

4.3.4 Команда «Ping»

Для проверки связи между устройствами сети можно использовать данную команду. Она отправляет эхо-запросы указанному узлу сети и фиксирует поступающие ответы.

Формат команды:

ping *A.B.C.D*

Пример выполнения команды:

```
Router#ping 77.134.25.133
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 77.134.25.133, timeout is 2 seconds:
..!!!
Success rate is 60 percent (3/5)
```

Каждый ICMP-пакет, на который был получен ответ, обозначается восклицательным знаком, каждый потерянный пакет – точкой.

4.4 Контекст пользователя

4.4.1 Команда «Enable»

Выполнение конфигурационных или управляющих команд требует вхождения в привилегированный режим, используя данную команду.

Пример выполнения команды:

```
Router>enable
Router#
```

При вводе команды маршрутизатор перешел в привилегированный режим. Для выхода из данного режима используется команда disable или exit.

Также следует отметить, что в данном контексте можно пользоваться командой show для просмотра некоторой служебной информации.

4.5 Контекст конфигурирования маршрутизации

4.5.1 Команда «Network»

Данной командой указывают адреса сетей, которые будут доступны данному маршрутизатору.

Формат команды:

network *A.B.C.D* , где A.B.C.D – адрес сети

Пример выполнения команды:

Router(config-router)#network *192.168.3.0*

Данная команда означает, что пакеты, направленные в подсеть 192.168.3.0 будут отправляться через данный шлюз.

5 Лабораторные работы

5.1 Лабораторная работа №1. Знакомство с симулятором Cisco Packet Tracer

Целью данной лабораторной работы является знакомство с симулятором Cisco Packet Trace и получение базовых навыков по работе с ним.

Задание:

- Спроектировать простейшую сеть;
- Ознакомиться с утилитой Ping и запустить ping-процесс.

Рекомендации к выполнению:

Сеть, которую должен спроектировать студент, изображена на рис.6.1.

Как известно, локальная вычислительная сеть – это компьютерная сеть, покрывающая обычно относительно небольшую территорию или небольшую группу зданий. В нашем случае это всего-навсего 6 рабочих станций, определенным образом связанных между собой. Для этого мы будем использовать сетевые концентраторы (хабы) и коммутаторы (свичи).

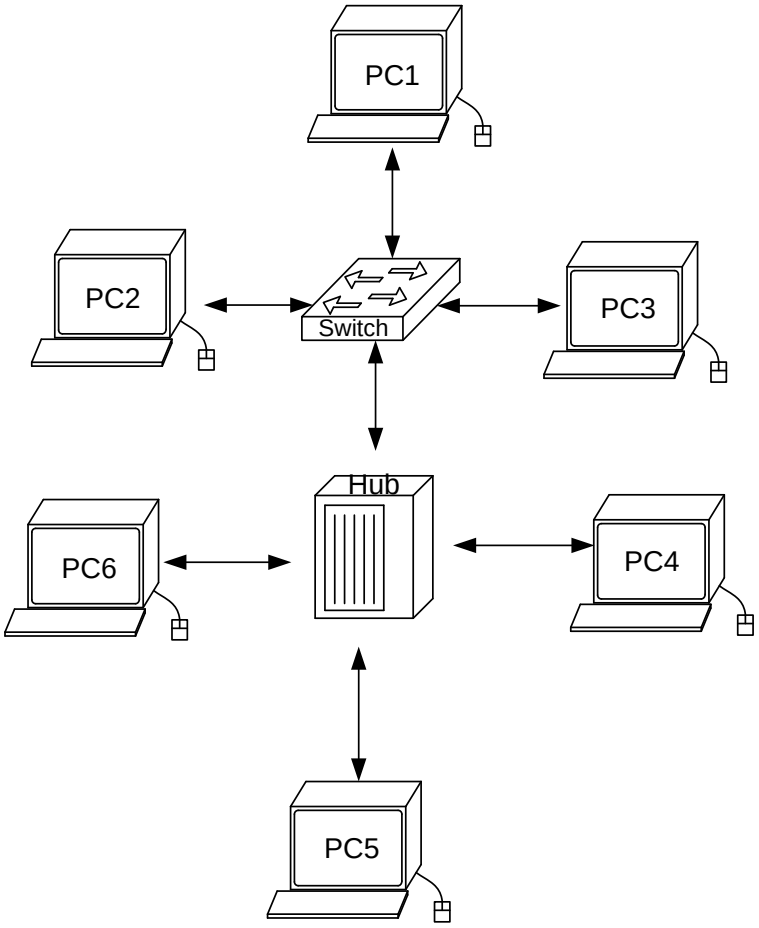


Рис.5.1. Проектируемая сеть

1. В нижнем левом углу Packet Tracer 4.0 выбираем устройства «Сетевые коммутаторы», и, в списке справа, выбираем коммутатор 2950-24,нажимая на него левой кнопкой мыши, вставляем его в рабочую область. Так же поступает с «Сетевым концентратором (Hub-PT)» и «Рабочими станциями (PC-PT)».

2. Далее необходимо соединить устройства, как показано на рис.1, используя соответствующий интерфейс. Для упрощения выбираем в нижнем левом углу Packet Tracer 4.0 «Тип связи» и указываем «Автоматически выбрать тип соединения»: нажимая на данный значок левой кнопкой мыши, затем нажимаем на необходимое нам устройство, и соединяем с другим все тем-же нажатием.

3. Далее идет самый важный этап – настройка. Так как мы используем устройства, работающие на начальных уровнях сетевой модели OSI (коммутатор на 2ом, концентратор – на 1ом), то их настраивать не надо. Необходима лишь настройка рабочих станций, а именно: IP-адреса, маски подсети, шлюза.

Ниже приведена настройка лишь одной станции (PC1) – остальные настраиваются аналогично.

Производим двойной щелчок по нужной рабочей станции, в открывшемся окне выбираем вкладку Рабочий стол, далее – Конфигурация интерфейса, и производим соответствующую настройку:

IP-адрес. Как известно, в локальных сетях, основанных на протоколе IP, могут использоваться следующие адреса:

• 10.0.0.0—10.255.255.255;

• 172.16.0.0—172.31.255.255;

• 192.168.0.0—192.168.255.255.

Поэтому выбираем IP-адрес из данных диапазонах, например 192.168.0.1

Важно! IP-адреса всех рабочих станций должны находиться в одной и той-же подсети (то есть из одного диапазона), иначе процесс ping не выполнится.

Маска подсети. Значение подставится автоматически, когда будет введен IP-адрес.

Шлюз. Поле можно не заполнять.

4. Когда настройка завершена, можно переходить ко второй части работы – к запуску ping-процесса. Например, запускать его будем с PC5 и проверять наличие связи с PC1.

Важно! Студент сам может выбрать, откуда ему запускать ping-процесс, главное, чтобы выполнялось условие: пакеты должны обязательно пересылаться через коммутатор и концентратор.

Для этого производим двойной щелчок по нужной рабочей станции, в открывшемся окне выбираем вкладку «Рабочий стол», далее – «Командная строка». Нам предлагают ввести команду, что мы и делаем:

```
PC>ping 192.168.0.1
```

и жмем клавишу Enter. Если все настроено верно, то мы увидим следующую информацию:

```
Pinging 192.168.0.1with 32 bytes of data:
Reply from 192.168.0.1: bytes=32 time=183ms TTL=120
Reply from 192.168.0.1: bytes=32 time=90ms TTL=120
Reply from 192.168.0.1: bytes=32 time=118ms TTL=120
Reply from 192.168.0.1: bytes=32 time=87ms TTL=120
Ping statistics for 192.168.0.1:
   Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
   Minimum = 87ms, Maximum = 183ms, Average = 119ms
PC>
```

Это означает, что связь установлена, и данный участок сети работает исправно.

5. В Packet Tracer предусмотрен режим моделирования, в котором подробно описывается и показывается, как работает утилита Ping. Поэтому необходимо перейти в данный режим, нажав на одноименный значок в нижнем левом углу рабочей области, или по комбинации клавиш Shift+S. Откроется «Панель моделирования», в которой будут отображаться все события, связанные с выполнения ping-процесса.

Теперь необходимо повторить запуск ping-процесса. После его запуска можно сдвинуть «Панель моделирования», чтобы на схеме спроектированной сети наблюдать за отправкой/приемкой пакетов.

Кнопка «Автоматически» подразумевает моделирование всего ping-процесса в едином процессе, тогда как «Пошагово» позволяет отображать его пошагово.

Чтобы узнать информацию, которую несет в себе пакет, его структуру, достаточно нажать правой кнопкой мыши на цветной квадрат в графе «Информация».

Моделирование прекращается либо при завершении ping-процесса, либо при закрытии окна «Редактирования» соответствующей рабочей станции.

5.2 Лабораторная работа №2. Основы работы с интерфейсом оборудования Cisco.

Целью данной лабораторной работы является получение базовых навыков по работе с командным интерфейсом коммутаторов Cisco. Рассматриваются приемы первичной настройки коммутаторов, обеспечения их защищенности и доступности для управления.

Новые приобретаемые навыки в работе с оборудованием Cisco:

- Изменение имени оборудования (hostname);
- Вход в привилегированный режим (enable);
- Вход в режим конфигурации настроек (configure terminal);
- Вход в режим конфигурирования линий (консоль, терминальные подключения) (line?);
- Вход в режим конфигурирования интерфейсов виртуальный сетей (interface VLAN ?);
- Задание пароля для перехода в привилегированный режим (enable secret?);

- Задание ip-адреса для интерфейса виртуальной сети коммутатором (ip address ?);
- Сохранение текущей конфигурации (copy running-config startup-config);
- Просмотр текущей работающей конфигурации (show running-config);
- Просмотр сохраненной конфигурации (show startup-config);
- Настройка ip-адресов персональных компьютеров (winipcfg, ipconfig ?);
- Выявление достижимости персональных компьютеров и коммутаторов в сети (ping?);
- Просмотр записей arp-таблицы персональных компьютеров (arp).

Схема сети:

- Коммутаторы S1, S2, S3 (3 шт.);
- Персональные компьютеры PC1, PC2, PC3, PC4 (4 шт.);
- Схема сети представлена на рис.6.2.

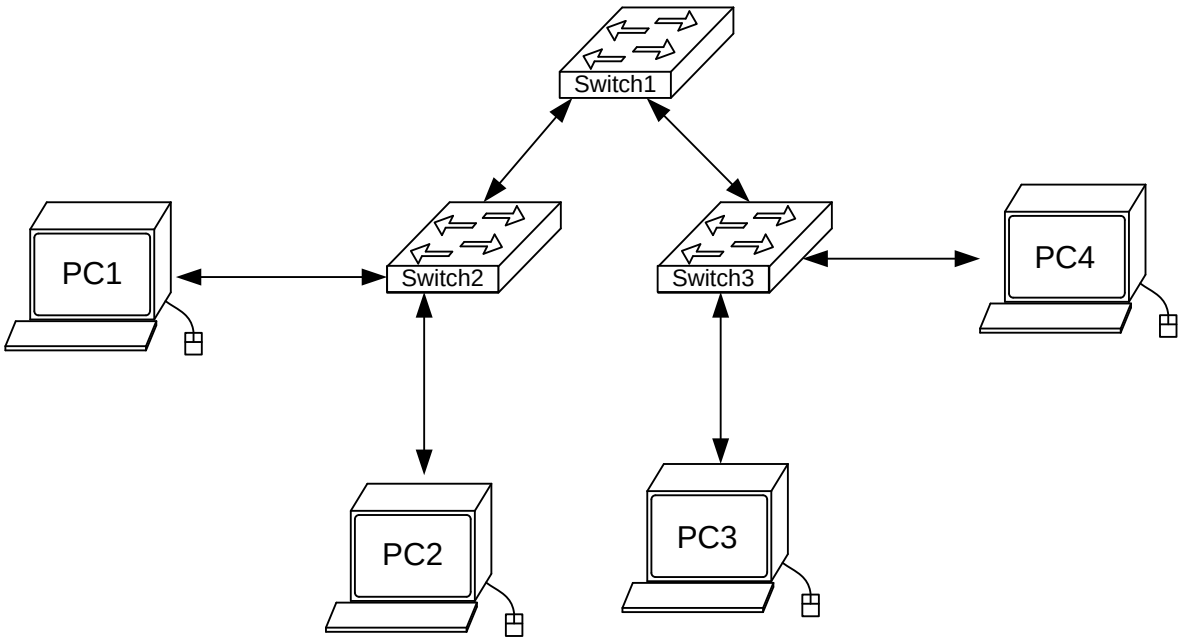


Рис.5.2. Схема сети

Задание:

- Изменить имя коммутаторам Cisco;

- Обеспечить парольный доступ к привилегированному режиму на коммутаторах;
- Задать ip-адреса и маски коммутаторам (172.16.1.11/24, 172.16.1.12/24, 172.16.1.13/24);
- Задать ip-адреса и маски сетей персональным компьютерам. (172.16.1.1/24, 172.16.1.2/24, 172.16.1.3/24, 172.16.1.4/24);
- Убедиться в достижимости всех объектов сети по протоколу IP;
- Переключившись в «Режим симуляции» (описанном в методических указаниях к предыдущей лабораторной работе) рассмотреть и пояснить процесс обмена данными по протоколу ICMP между устройствами (выполнив команду Ping с одного компьютера на другой), пояснить роль протокола ARP в этом процессе. Детальное пояснение включить в отчет.

Структура отчета по работе:

- Титульный лист;
- Задание;
- Схема сети;
- Ход работы:
    - Данный раздел состоит из последовательного описания значимых выполняемых шагов (с указанием их сути) и копий экранов (должна быть видна набранная команда и реакция системы, если она есть).
- Выводы.

5.3 Лабораторная работа №3. Настройка статической маршрутизации на оборудовании Cisco.

Целью данной лабораторной работы является изучение процессов настройки статических маршрутов на маршрутизаторах Cisco.

Новые приобретаемые навыки в работе с оборудованием Cisco:

- Синхронизация времени для последовательных сетевых интерфейсов (clock rate ?);
- Задание статических маршрутов и маршрутов «по умолчанию» (ip route ?);
- Просмотр созданной таблицы маршрутов (show ip route ?).

Схема сети:

- Коммутаторы S1, S2, S3 (3 шт.);
- Маршрутизаторы R1, R2, R3 (3 шт.);
- Персональные компьютеры C1, C2, C3 (3 шт.);
- Схема сети представлена на рис.6.3.

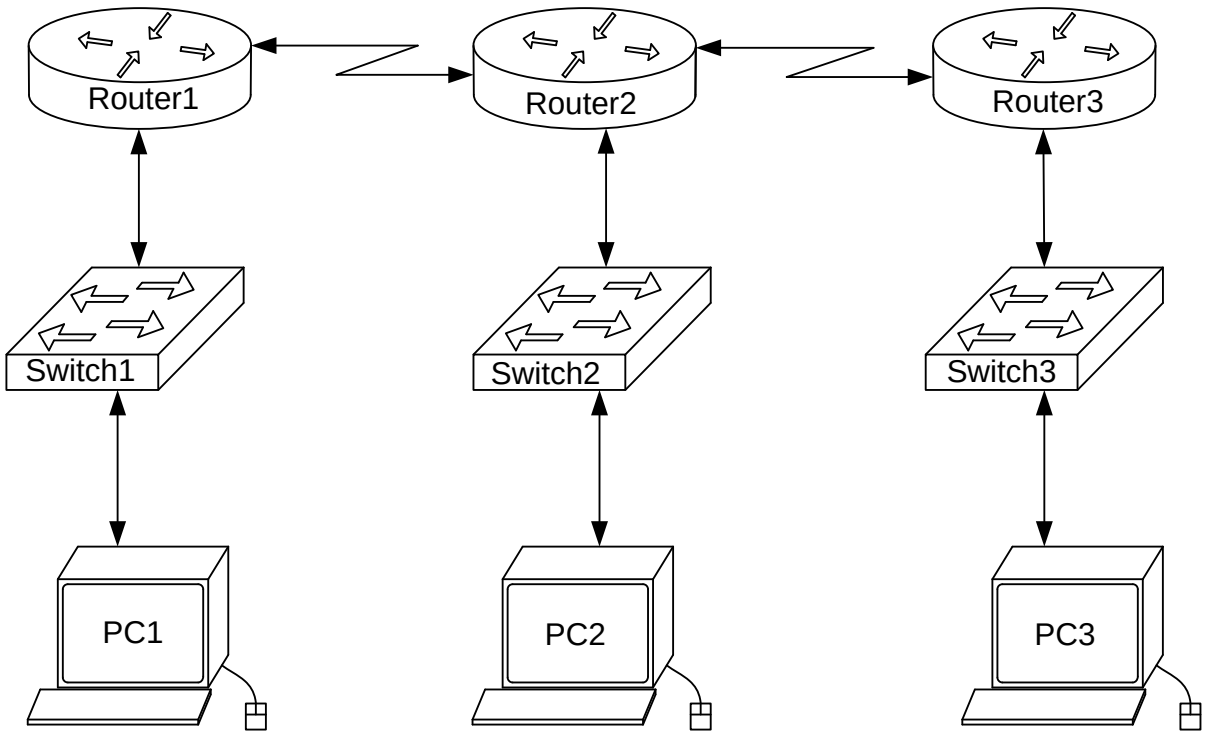


Рис.5.3. Схема сети

Задание:

- Задать IP адреса сетевым интерфейсам маршрутизаторов, интерфейсам управления коммутаторов и сетевым интерфейсам локальных компьютеров;
- Установить связь на физическом и канальном уровнях между соседними маршрутизаторами по последовательному сетевому интерфейсу;
- Добиться возможности пересылки данных по протоколу IP между соседними объектами сети (C1-S1, C1-R1, S1-R1, R1-R2, R2-S2, R2-C2, и т.д.);
- Настроить на маршрутизаторе R2 статические маршруты к сетям локальных компьютеров C1, C3
- Настроить на маршрутизаторах R1, R3 маршруты «по умолчанию» к сетям локальных компьютеров C2-C3 и C1-C2 соответственно;
- Добиться возможности пересылки данных по протоколу IP между любыми объектами сети (ping);
- Переключившись в «Режим симуляции» рассмотреть и пояснить процесс обмена данными по протоколу ICMP между устройствами (выполнив команду Ping с одного компьютера на другой), пояснить роль протокола ARP в этом процессе. Детальное пояснение включить в отчет.

Структура отчета по работе:

- Титульный лист;
- Задание;
- Топологическая схема сети:
    - Указать на схеме наименования узлов сети, адреса и типы сетевых интерфейсов.
- Ход работы:
    - Данный раздел состоит из последовательного описания значимых выполняемых шагов (с указанием их сути) и копий экранов

(должна быть видна набранная команда и реакция системы, если она есть).

- Конфигурации оборудования:
  - Привести значимые фрагменты конфигурационных файлов (startup—config) для коммутаторов и маршрутизаторов Cisco, пояснить значение команд.
- Выводы.

5.4 Лабораторная работа №4. Настройка протоколов маршрутизации RIP на оборудовании Cisco.

Целью данной лабораторной работы является настройка протоколов динамической маршрутизации на оборудовании Cisco.

Новые приобретаемые навыки в работе с оборудованием Cisco:

- Включение на маршрутизаторе поддержки протокола RIP (router rip);
- Настройка протокола RIP на поддержку маршрутизации требуемых сетей (network?);
- Просмотр таблицы маршрутизации (show ip route);
- Просмотр работающих протоколов маршрутизации (show ip protocols).

Схема сети:

- Коммутаторы S1, S2, S3 (3 шт.);
- Маршрутизаторы R1, R2, R3 (3 шт.);
- Персональные компьютеры C1, C2, C3 (3 шт.);
- Схема сети представлена на рис.6.4.

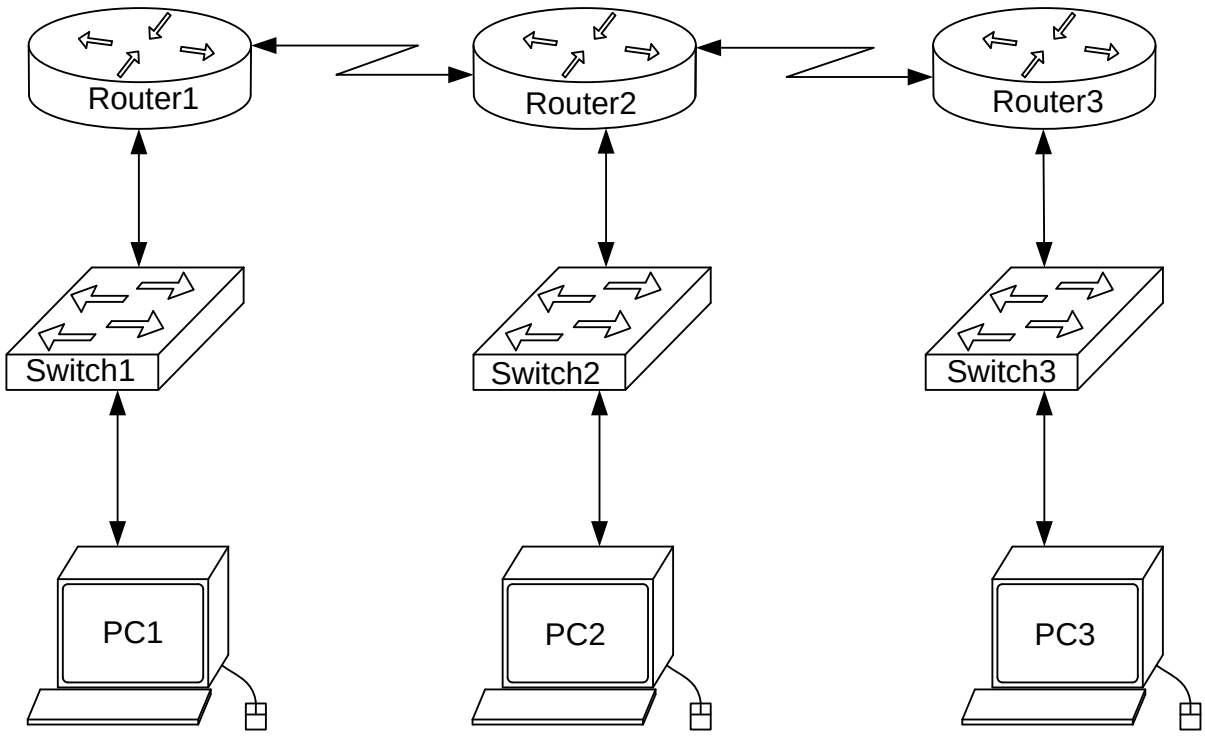


Рис.5.4. Схема сети

Задание:

- Задать IP адреса сетевым интерфейсам маршрутизаторов, интерфейсам управления коммутаторов и сетевым интерфейсам локальных компьютеров;
- Установить связь на физическом и канальном уровнях между соседними маршрутизаторами по последовательному сетевому интерфейсу;
- Добиться возможности пересылки данных по протоколу IP между соседними объектами сети (C1-S1, C1-R1, S1-R1, R1-R2, R2-S2, R2-C2, и т.д.);
- Выявить невозможность пересылки данных по протоколу IP между удаленными объектами сети;
- Просмотреть существующую таблицу маршрутизации;
- Включить поддержку протокола RIP на всех маршрутизаторах сети;
- Подключить к протоколу RIP требуемые сети;
- Просмотреть обновленную таблицу маршрутизации;

- Посмотреть список протоколов маршрутизации работающих на узлах сети;
- Удостовериться в возможности пересылки данных по протоколу IP между любыми объектами сети.

Структура отчета по работе:

- Титульный лист;
- Задание;
- Топологическая схема сети:
    - Указать на схеме наименования узлов сети, адреса и типы сетевых интерфейсов.
- Ход работы:
    - Данный раздел состоит из последовательного описания значимых выполняемых шагов (с указанием их сути) и копий экранов (должна быть видна набранная команда и реакция системы, если она есть).
- Конфигурации оборудования:
    - Привести значимые фрагменты конфигурационных файлов (startup—config) для коммутаторов и маршрутизаторов Cisco, пояснить значение команд.
- Выводы.

5.5 Лабораторная работа №5. Применение списков доступа на оборудовании Cisco.

Целью данной лабораторной работы является настройка стандартных списков доступа на маршрутизаторах Cisco и знакомство с расширенными списками доступа.

Новые приобретаемые навыки в работе с оборудованием Cisco:

- Сопоставление интерфейсу маршрутизатора некоторой группы доступа (ip access-group ?);
- Создание списков доступа позволяющих или препятствующих передачи данных между узлами сети (access-list ?).

Схема сети:

- Коммутаторы S1, S2, S3 (3 шт.);
- Маршрутизаторы R1, R2, R3 (3 шт.);
- Персональные компьютеры PC1, PC2, PC3 (3 шт.);
- Схема сети представлена на рис.6.5.

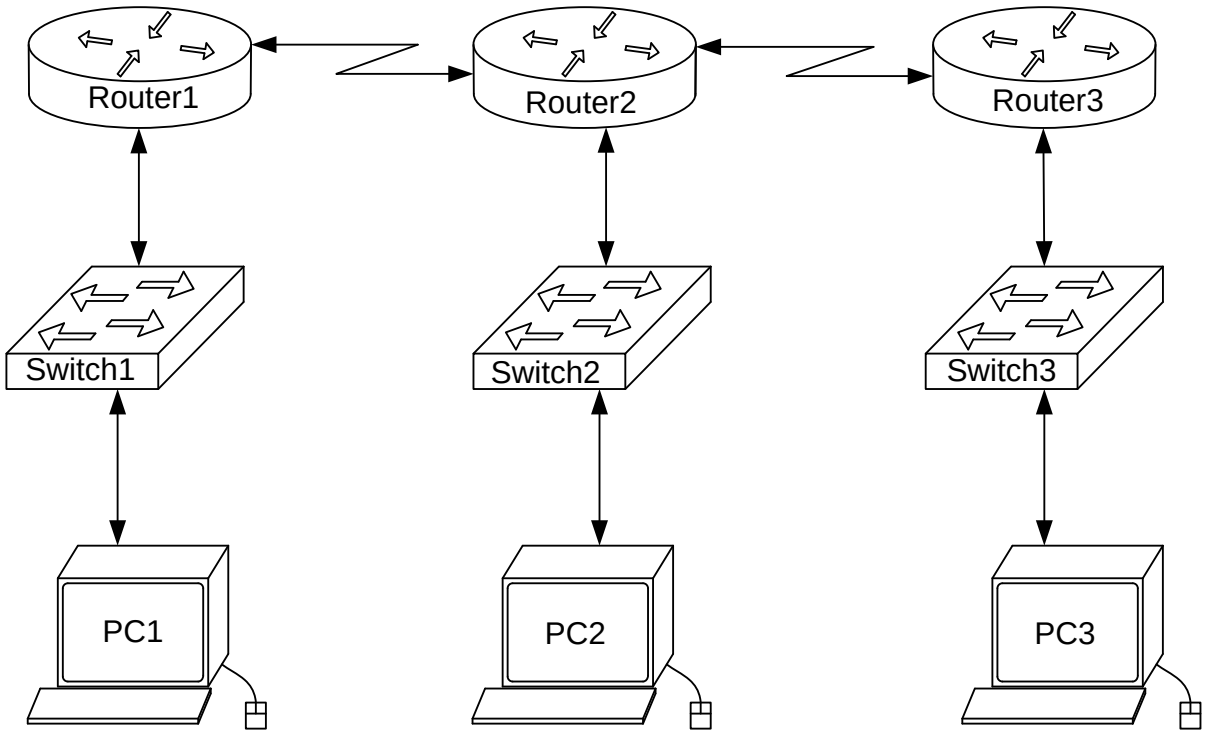


Рис.5.5. Схема сети

Задание:

- Задать всем узлам сети IP адреса;
- Настроить динамическую или статическую маршрутизацию всеми узлами сети;

- Выявить возможность пересылки данных по протоколу IP между любыми объектами сети;
- Разработать и применить на маршрутизаторах списки доступа:
    - Запрещающие маршрутизаторам R0 и R2 обмениваться ICMP-пакетами по последовательному сетевому интерфейсу;
    - Запрещающие компьютерам PC0 и PC1 обмениваться ICMP-пакетами по интерфейсу Ethernet.
- Переключившись в «Режим симуляции» рассмотреть и пояснить процесс обмена данными по протоколу RIP (в случае динамической маршрутизации) между устройствами (выполнив команду Ping с одного компьютера на другой). Детальное пояснение включить в отчет.

Структура отчета по работе:

- Титульный лист;
- Задание;
- Топологическая схема сети:
    - Указать на схеме наименования узлов сети, адреса и типы сетевых интерфейсов.
- Ход работы:
    - Данный раздел состоит из последовательного описания значимых выполняемых шагов (с указанием их сути) и копий экранов (должна быть видна набранная команда и реакция системы, если она есть).
- Конфигурации оборудования;
    - Привести значимые фрагменты конфигурационных файлов (startup—config) для коммутаторов и маршрутизаторов Cisco, пояснить значение команд.
- Выводы.

5.6 Лабораторная работа №6. Проектирование сети SOHO

Целью данной лабораторной работы является применение навыков проектирования и реализации сети, для типовой задачи класса SOHO.

Новые приобретаемые навыки в работе с оборудованием:

- Самостоятельность принятия решений по архитектуре сети
- Применение знаний для реализации прикладной задачи

Постановка задачи:

Организация располагается на 3 х этажах здания. Парк компьютеров составляет 60 машин. На первом этаже существует гостевое помещение, в котором необходим гостевой Wi-Fi доступ в интернет для 10-15 человек единовременно. В сети существуют несколько серверов – осуществляющих следующие сервисы: HTTP, DNS, FTP, DHCP, EMAIL, общие папки «Share». Службы HTTP, FTP и EMAIL. Должны быть доступны как изнутри сети, так и из «интернета».

Бухгалтерия насчитывает 5-8 машин. Бухгалтерия работает с «бухгалтерским» сервером скрыто от остальных.

Для проверки работоспособности доступа «Интернет» - использовать Сервер, подключенный к WAN со следующими службами: DNS, HTTP.

Задание:

- Задать всем узлам сети IP адреса;
- Настройка сетевых интерфейсов и сервисов
- Выявить возможность пересылки данных по протоколу IP между необходимыми объектами сети;
- Проверить работоспособность заданных сервисов в рамках поставленной задачи
- Исследовать движения пакетов созданных сервисов.

Структура отчета по работе:

- Титульный лист;
- Задание;
- Топологическая схема сети:
  - Указать на схеме наименования узлов сети, адреса и типы сетевых интерфейсов.
- Ход работы:
  - Данный раздел состоит из последовательного описания значимых выполняемых шагов (с указанием их сути) и копий экранов (должна быть видна набранная команда и реакция системы, если она есть).
- Конфигурации оборудования;
  - Привести значимые фрагменты конфигурационных файлов (startup—config) для коммутаторов и маршрутизаторов Cisco, пояснить значение команд.
- Выводы.

Литература

19. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологи, протоколы: учебник для вузов. 4-е изд. – СПб.: Питер, 2013. – 944с.
20. Одом, Уэнделл. Официальное руководство Cisco по подготовке к сертифицированным экзаменам CCENT/CCNA ICND1. 2-е изд. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2011 – 672с.
21. Таненбаум Э. Компьютерные сети. 4-е изд. – СПб.: Питер, 2010. – 992с.

22. Синди Фейт. TCP/IP. Архитектура, протоколы, реализация (включая IPv6 и IP Security)- 2-е изд. Издательство: ЛОРИ, 2009. - 424 с.

23. Дансмор Бред, Скандьер Тоби. Справочник по телекоммуникационным технологиям. : Пер. с. Анг. – М. : Издательский дом «Вильямс», 2004. – 640 с.

Приложение 1

Справочник по основным командам

show version

Показывает наименование модели циски, версию IOS, название и местоположение двоичного образа загрузки IOS (обычно во flash: сжатый файл с названием cMMMM-D-EE.NNN-O.F.bin, где MNO цифры, DEF буквы), распределение и размер ОЗУ, размер NVRAM, размер flash, содержимое конфигурационного регистра.

show startup-config

Показывает содержимое конфигурации, которая применяется при загрузке. Можно скопировать эти данные в буфер обмена и сохранить в файл в качестве бэкапа конфигурации. Этот файл потом можно просто вставить (с небольшими оговорками) из буфера обмена в экран консоли, дать команду wr mem, и этим восстановить конфигурацию (многие программы, автоматически сохраняющие и обновляющие конфигурацию, применяют как раз такой метод).

copy startup-config running-config

Отменяет все сделанные (если были) изменения в конфигурации. То же самое произойдет, если выключить/включить питание (перезагрузить устройство).

copy running-config startup-config

Сохраняет в энергонезависимой памяти все изменения, сделанные в конфигурации. Полный аналог команды write или write memory.

write

Сохраняет в энергонезависимой памяти все изменения, сделанные в конфигурации. Полный аналог команды write memory или copy running-config startup-config.

show flash

Показывает размер, свободное место и содержимое (в виде списка) энергонезависимой памяти, которая работает с точно так же, как диск. На

этом диске хранятся файлы, с которых записана IOS и конфигурация циски (startup-config и другие). Файлами можно манипулировать командами IOS.

terminal monitor

Переключает вывод debug-информации с консольного порта (RS232) на консоль, подключенную через сетевой интерфейс.

(no) service password-encryption

Команда, которая показывает пароли enable в конфиге в (открытом)закрытом виде

(no) logging console

Команда, которая (выключает)включает вывод сообщений системного журнала на консоль (RS232)

logging console <0-7>

Команда, включает вывод сообщений системного журнала на консоль (RS232) определённого уровня (0 меньше всего сообщ. - emergencies System is unusable, 7 - больше всего сообщений - Debugging messages)

loggig buffered

Указание сохранять сообщения в ОЗУ для последующего ознакомления

show logging

Вывод сообщений из ОЗУ

show flash: all

Показывает статус flash - сколько занято, свободно, контрольные суммы, сколько банков и их параметры, тип микросхем памяти.

show interfaces

Одна из наиболее часто используемых команд, показывает количество, параметры, состояние интерфейсов и ещё много разной информации.

show vlan

Показать существующие vlan и привязку к ним физических интерфейсов.

erase nvram

Очистка конфигурации (startup-config и другая информация), полный сброс энергонезависимой памяти.

configure <memory|network|overwrite-network|terminal>

Вход в один из разновидностей режима configure - изменение конфигурации. Наиболее часто используемый режим configure terminal (подключение как через консольный порт RS232, так и по сети через telnet или ssh).

end

Полный выход из режима configure. Тот же эффект дает Ctrl-Z.

exit

Шаг назад по дереву конфигурирования (например, выход из реж. конфигурирования одного из интерфейсов).

no vlan n

Удалить vlan n.

(no) shutdown

Административно (включить) выключить сетевой интерфейс.

show vtp status

Показать конфигурацию режима VTP.

vtp mode <server|client>

Включить требуемый режим работы VTP.

show debugging

Показать накопленную (в памяти) статистику отладки.

undebug all

Полностью выключить отладку.

traceroute aaa.bbb.ccc.ddd

Аналог tracert aaa.bbb.ccc.ddd - показать маршрут до указанного IP.

show process cpu

Показать статистику загрузки процессора (в том числе и каждой задачей).

show process cpu history

Показать статистику загрузки процессора с временнЫми графиками.

Who

Показать сеансы администраторов, залогинившихся в терминал циски.

ssh -v 2 -l root a.b.c.d

Подсоединиться к Knopix по SSH версии 2.

no banner login

Удаляет из конфига все строки banner login (приветствие при логине).

show interfaces port-channel n

Показывает состояние канала портов под номером n, какие порты туда входят.

show ip eigrp neighbors

Показывает EIGRP-соседей, какими интерфейсами с ними контакт, номер EIGRP-процесса.

show ip eigrp interfaces

Показывает список интерфейсов, вовлеченных в EIGRP, номер EIGRP-процесса.

show ip eigrp traffic

show ip eigrp topology

Показывает статистику работы EIGRP, номер EIGRP-процесса.

snmp-server community <строка_пароль> <RW / RO> [номер access-листа]

Команда вводится в режиме глобального конфигурирования. Настраивает доступ к внутреннему snmp-серверу для специального ПО (например, чтобы CiscoWorks Device Fault Manager мог собирать статистику о состоянии оборудования). Параметр <строка_пароль> представляет собой community-string, который используется для аутентификации при подключении. Если указать RW, то будет разрешен полный доступ (чтение и запись) в SNMP базу данных устройства (можно не только считывать состояние, но и менять параметры устройства), если RO, то доступ будет

только на чтение. Номер access-листа позволяет отфильтровать нежелательные подключения.

setup

Команда setup привилегированного режима запускает мастера первоначальной настройки.

terminal history size n

Команда, меняющая количество запоминаемых ранее введенных команд (n max 256).

telnet IP-адрес

Команда позволяет подключиться к другой циске. <Ctrl+Shift+6> <x> позволяет приостановить сеанс Telnet (не разрывая его) и вернуться к собственной командной строке устройства. Команда disconnect без параметров позволяет разорвать последнее приостановленное соединение, а resume без параметров возобновляет последнее приостановленное соединение.

show diag [номер слота]

Команда показывает подробную информацию о материнской плате устройства Cisco и/или об установленных в слоты адаптерах.

show environment

Команда на некоторых устройствах (чаще дорогих и продвинутых) показывает состояние вентиляторов и температуру устройства, иногда значение питающих напряжений.

show ip interface

Команда показывает подробную информацию об интерфейсах в контексте протокола IP.

show ip sockets

Команда показывает открытые порты и активные соединения устройства Cisco.

show ip traffic

Команда показывает подробную инфо по трафику протоколов IP (много всего, в том числе количество пакетов broadcast и multicast), ICMP, TCP, BGP, IP-EIGRP, PIMv2, IGMP, UDP, OSPF, ARP и об ошибках.

show line

Команда показывает инфо о линиях - интерфейсах специального типа, предназначенных для администрирования. Обычно это последовательная консоль (CTY, console 0 или линия 0), AUX (обычно линия 1) и консоли Telnet (например, VTY 0-181 или линии 18-199).

show line summary

Команда показывает быструю сводку о статистике использования всех линий.

show sessions

Команда показывает информацию приостановленных сессиях Telnet.

show snmp

Команда показывает статистику протокола SNMP (полезно при настройке и проверке работы протокола).

show tcp

Команда показывает подробную статистику о всех открытых соединениях с устройством Cisco

send *

Команда позволяет отсылать сообщение в консоль всех залогиненных пользователей на устройстве.

verify flash:имя_файла_IOS

Команда позволяет проверить целостность файла (проверяются контрольные суммы). Полезно выполнить после копирования IOS во флеш (например, при обновлении IOS-а).

clear ip nat translation *

Очистка таблицы NAT, обычно применяемая при смене правил NAT.

(config)# do команда

Очень полезная команда **режима конфигурирования**, которая позволяет вводить команды **обычного режима** (например, do show running-config). Правда, в команде do уже не работает автозавершение команды клавишей **Tab** и подсказка по вводу **?**.

Приложение 2

Физическое комплектование оборудования в Cisco Packet Tracer

Физическое комплектование заключается в дополнении его модульных составляющих и последующей их настройке.

Физическая комплектация оборудования в эмуляторе

Переносим на рабочее поле роутер Cisco 1841. По сравнению с 1700 линейкой в 5 раз увеличили производительность. Кликом на роутере открываем его **физическую конфигурацию.**

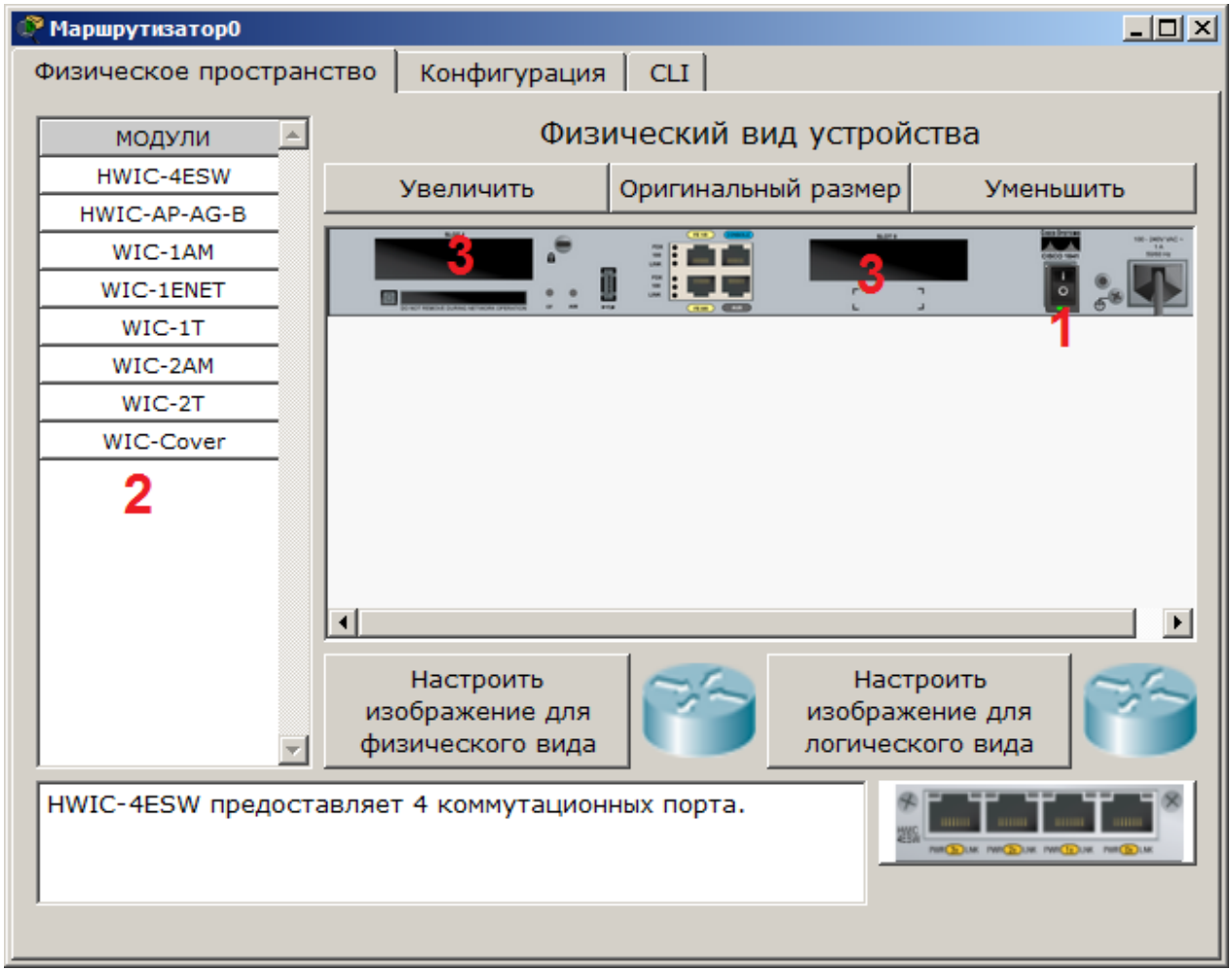


Рис. П2.1Физическое пространство маршрутизатора

Слева, как мы видим, список модулей (цифра 2), которыми можно укомплектовать данный роутер. Видите 2 пустоты в нём (цифра 3)? В них можно вложить эти модули. Разумеется, все эти дела нужно производить при

выключенном питании (цифра 1). А что вы думали? Всё по-настоящему, чтобы сразу приучались.

Что нужно знать о модулях WIC (HWIC, VWIC)

Перед тем, как я распишем, какие дополнительные модули предлагаются для маршрутизаторов, введем два понятия:

- WIC — WAN interface card. the first original models.
- HWIC- high-speed wan interface card- the evolution of wic that is now in use on the ISR routers.
- VIC — voice interface card, support voice only.
- VIC2 — evolution of the above
- VWIC — voice and wan interface card. An E1/T1 card that can be user for voice or data.
- VWIC2 — evolution of the above

Иначе говоря, это платы расширения, увеличивающие функционал устройства. Как например для компьютера есть платы, подключаемые к PCI-шине (TV-тюнеры, звуковые карты, USB-разветвители, сетевые карты), так и здесь. Вообще, устройство Cisco — это тот же системный блок со своей операционной системой и многими сетевыми картами, который может делать что-то только с сетью, но зато то, как он это делает — выше всяких похвал.

А теперь подробнее о тех модулях, что нам предоставляет Cisco Packet Tracer

- **HWIC — 4ESW** — высокопроизводительный модуль с 4-мя коммутационными портами Ethernet под разъем RJ-45. Позволяет сочетать в маршрутизаторе возможности коммутатора.
- **HWIC-AP-AG-B** — это высокоскоростная WAN-карта, обеспечивающая функционал встроенной точки доступа для роутеров

линейки Cisco 1800 (модульных), Cisco 2800 и Cisco 3800. Данный модуль поддерживает радиоканалы Single Band 802.11b/g или Dual Band 802.11a/b/g.

- **WIC-1AM** включает в себя два разъема RJ-11 (телефоннка), используемых для подключения к базовой телефонной службе. Карта использует один порт для соединения с телефонной линией, другой может быть подключен к аналоговому телефону для звонков во время простоя модема.
- **WIC-1ENET** — это однопортовая 10 Мб/с Ethernet карта для 10BASE-T Ethernet LAN.
- **WIC-1T** предоставляет однопортовое последовательное подключение к удаленным офисам или устаревшим серийным сетевым устройствам, например SDLC концентраторам, системам сигнализации и устройствам packet over SONET (POS).
- **WIC-2AM** содержит два разъема RJ-11, используемых для подключения к базовой телефонной службе. В WIC-2AM два модемных порта, что позволяет использовать оба канала для соединения одновременно.
- **WIC-2T**- 2-портовый синхронный/асинхронный серийный сетевой модуль предоставляет гибкую поддержку многих протоколов с индивидуальной настройкой каждого порта в синхронный или асинхронный режим. Применения для синхронной/асинхронной поддержки представляют:
    - низкоскоростную агрегацию (до 128 Кб/с);
    - поддержку dial-up модемов;
    - синхронные или асинхронные соединения с портами управления другого оборудования и передачу устаревших протоколов типа Bi-sync и SDLC.

- **WIC-Cover** — стенка для WIC слота, необходима для защиты электронных компонентов и для улучшения циркуляции охлаждающего воздушного потока.

Совместимость разных карт и модулей можно найти на сайте Cisco.

Как же комплектовать оборудование?

Итак, мы отключили питание, кликнув мышью на кнопке питания, перетащили модуль **4ESW** в свободный слот и включили питание. Подождали окончания загрузки роутера, и в конфигурации GUI можем увидеть появившиеся 4 новых интерфейса.

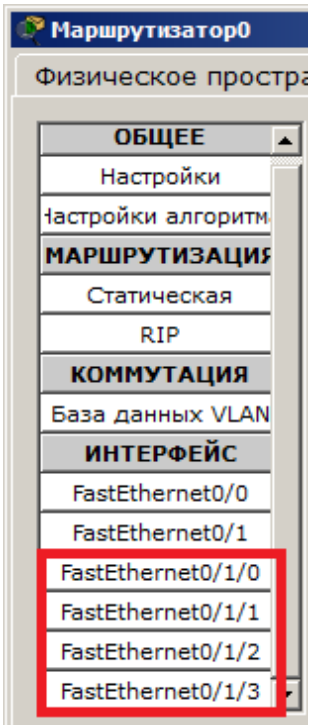


Рис. П2.2 Дополнительные интерфейсы после добавления модулей.

Остальные устройства комплектуются аналогично! Добавляются новые модули Ethernet (10/100/1000), оптоволоконные разъемы нескольких типов, адаптеры беспроводной сети. На рабочий компьютер есть возможность навесить например микрофон с наушниками, жесткий диск для хранения данных.

Рис. П2.3 Подключённые к компьютеру наушники и жесткий диск.

Приложение 3

Оборудование и линии связи в Cisco Packet Tracer

Маршрутизаторы

Рис.П3.1 Изображение маршрутизаторов

Для чего нужен маршрутизатор? Не такой простой вопрос, как кажется на первый взгляд. Маршрутизаторы используется для поиска оптимального маршрута передачи данных на основании специальных алгоритмов маршрутизации, например выбор маршрута (пути) с наименьшим числом транзитных узлов. Работают на сетевом уровне модели OSI.

Рис.П3.2 Изображение маршрутзатора 1841

Коммутаторы

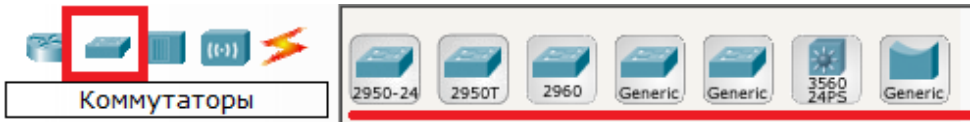


Рис.П3.3 Изображение коммутаторов

Коммутаторы — это устройства, работающие на канальном уровне модели OSI и предназначенные для объединения нескольких узлов в пределах одного или нескольких сегментах сети. Передаёт пакеты коммутатор на основании внутренней таблицы — таблицы коммутации,

следовательно трафик идёт только на тот MAC-адрес, которому он предназначается, а не повторяется на всех портах (как на концентраторе).

Рис.П3.4 Изображение коммутатора 2960-24TT

Концентраторы

Рис.П3.5 Изображение концентраторов

Это более «глупый» вариант устройства, объединяющего сетевые узлы. Просто напросто повторяет пакет, принятый на одном порту на всех остальных портах. Всё по технологии Ethernet. В настоящее время выпускаются очень редко. Никакой защиты. Просто этакий «тройник» как для силовой сети.

Беспроводные устройства

Рис.П3.4 Изображение беспроводных устройств

Название говорит само за себя. Беспроводные технологии Wi-Fi и сети на их основе. Включает в себя точки доступа.

Линии связи

Рис.П3.5 Изображение линий связи

А с помощью этих компонентов мы будем соединять узлы в единую схему. Здесь есть и автоматическое определение и консольный кабель и витая пара и оптоволокно. *Есть разница между прямым и кроссоверным кабелем, так что будьте внимательны.*

Конечные устройства

Рис. П3.6 Изображение конечных узлов

Здесь непосредственно конечные узлы, хосты, сервера, принтеры, телефоны и многое другое. Очень широкий перечень устройств.

Эмуляция Интернета

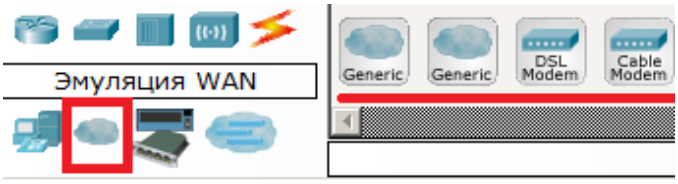


Рис.П3.7 Эмуляция WAN

А тут эмуляция глобальной сети. Модем DSL, «облако» и т.д.

Рис.П3.8 Пользовательские устройства и облако для многопользовательской работы

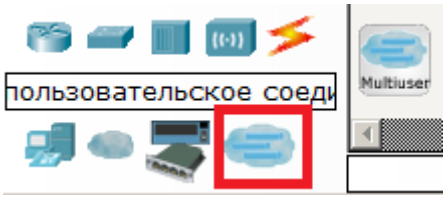


Рис.П3.9 Пользовательские соединения

Приложение 4

Cisco Packet Tracer настройки оборудования

Рассмотрим вопрос о создании дополнительных сервисов в Cisco Packet Tracer, для проверки работоспособности оборудования.

Обзор сервисов устройства Server

Рассмотрим псевдоустройство «Сервер». Найти его можно на вкладке «End Devices». SERVER представляет собой комплексное решение, вбирающее в себя основные функции (разумеется, только самый примитив) сервисов:

- **HTTP** – позволяет строить примитивные веб-странички и проверять прохождение пакетов на 80-ый порт сервера.
- **DHCP** – позволяет организовывать пулы сетевых настроек для автоматического конфигурирования сетевых интерфейсов.
- **TFTP** – простой протокол передачи файлов, годится для заливки и копирования конфигов и прошивок оборудования CISCO.
- **DNS** – позволяет организовать примитивную службу разрешения доменных имён.
- **SYSLOG** – позволяет организовать сбор сообщений, посылаемых различными устройствами на сервер логирования событий, будь то отключение питания, сбой безопасности и т.д.
- **AAA** – сервер авторизации;
- **NTP** – сервер синхронизации времени;
- **EMAIL** – почтовый сервер, для проверки элементарных почтовых правил;
- **FTP** – файловый сервер, полезная вещь для отладки различных ACL и загрузки прошивок;

Проверяем соединение с сервером

Итак, соединяем сервер с узлом через коммутатор, например. Можно и напрямую кабелем. Проверяем прохождение пакетов через PING:

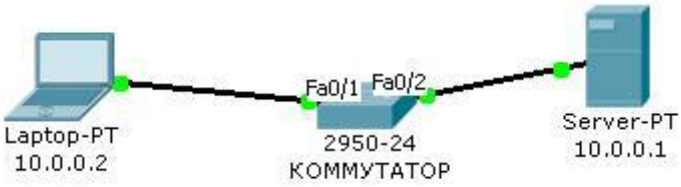


Рис.П4.1 Подключение сервера с ноутбуком через коммутатор

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1
Pinging 10.0.0.1 with 32 bytes of data:
Reply from 10.0.0.1: bytes=32 time=125ms TTL=128
Reply from 10.0.0.1: bytes=32 time=63ms TTL=128
Reply from 10.0.0.1: bytes=32 time=62ms TTL=128
Reply from 10.0.0.1: bytes=32 time=63ms TTL=128
Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 62ms, Maximum = 125ms, Average = 78ms
```

Создаем WEB-документы на сервере

Итак, создаём HTTP-сервер. Открываем вкладку «1» и создаём первую страничку, она у нас называется index.html «3». Обратите внимание, чтобы служба HTTP была включена –«2».

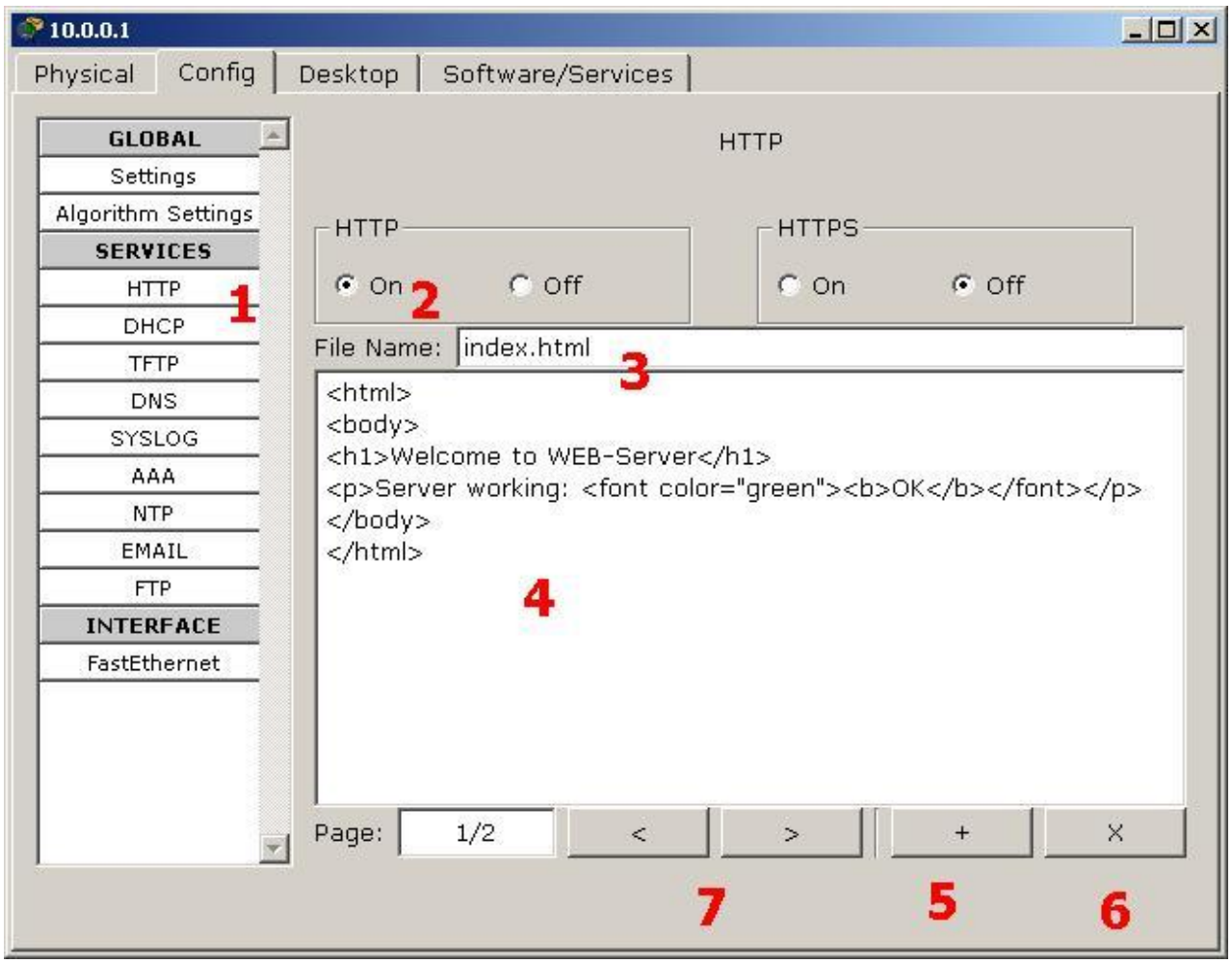


Рис.П4.2 Настройка сервера

В окно кода «4» вводим следующий текст для проверки (это простой HTML-документ):

```
<html>
<body>
<h1>Welcome to WEB-Server</h1>
<p>Server working: <font color=»green»><b>OK</b></font></p>
</body>
</html>
```

При желании можно добавить новую страницу «5» или удалить текущую «6». Переключение между страницами осуществляется кнопками «7».

Проверяем работоспособность HTTP-сервера

Для того, чтобы проверить работоспособность нашего сервера, открываем клиентскую машину (ноутбук 10.0.0.2) и на вкладке «Рабочий стол» открываем приложение Web Browser. После чего набираем адрес нашего WEB-сервера (пока только адрес, позднее с настройкой DNS можно будет набирать и доменное имя).

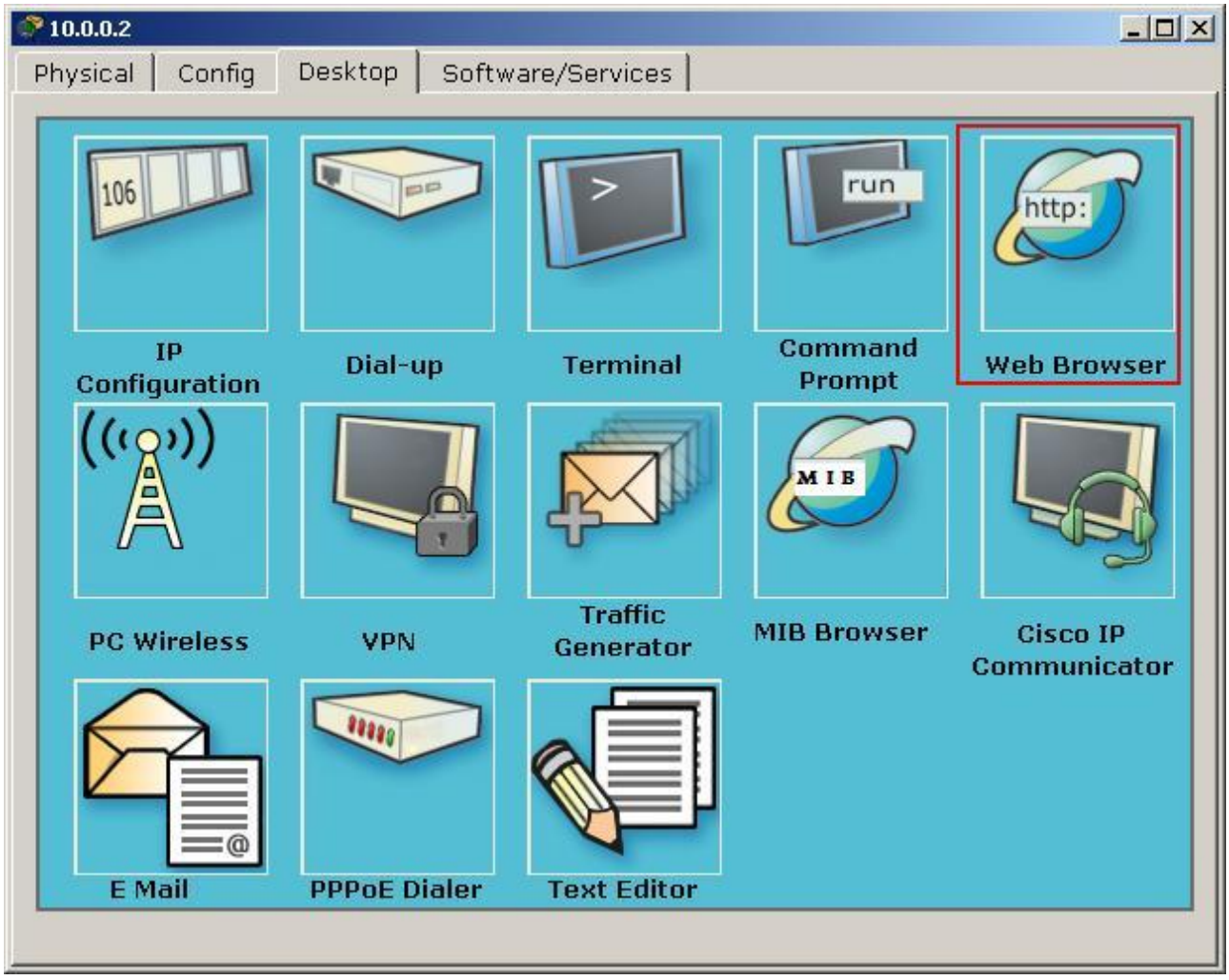


Рис.П4.3 Запускаем браузер на ноутбуке

И увидим следующую картину, что сигнализирует о том, что наш веб-сервер работает.

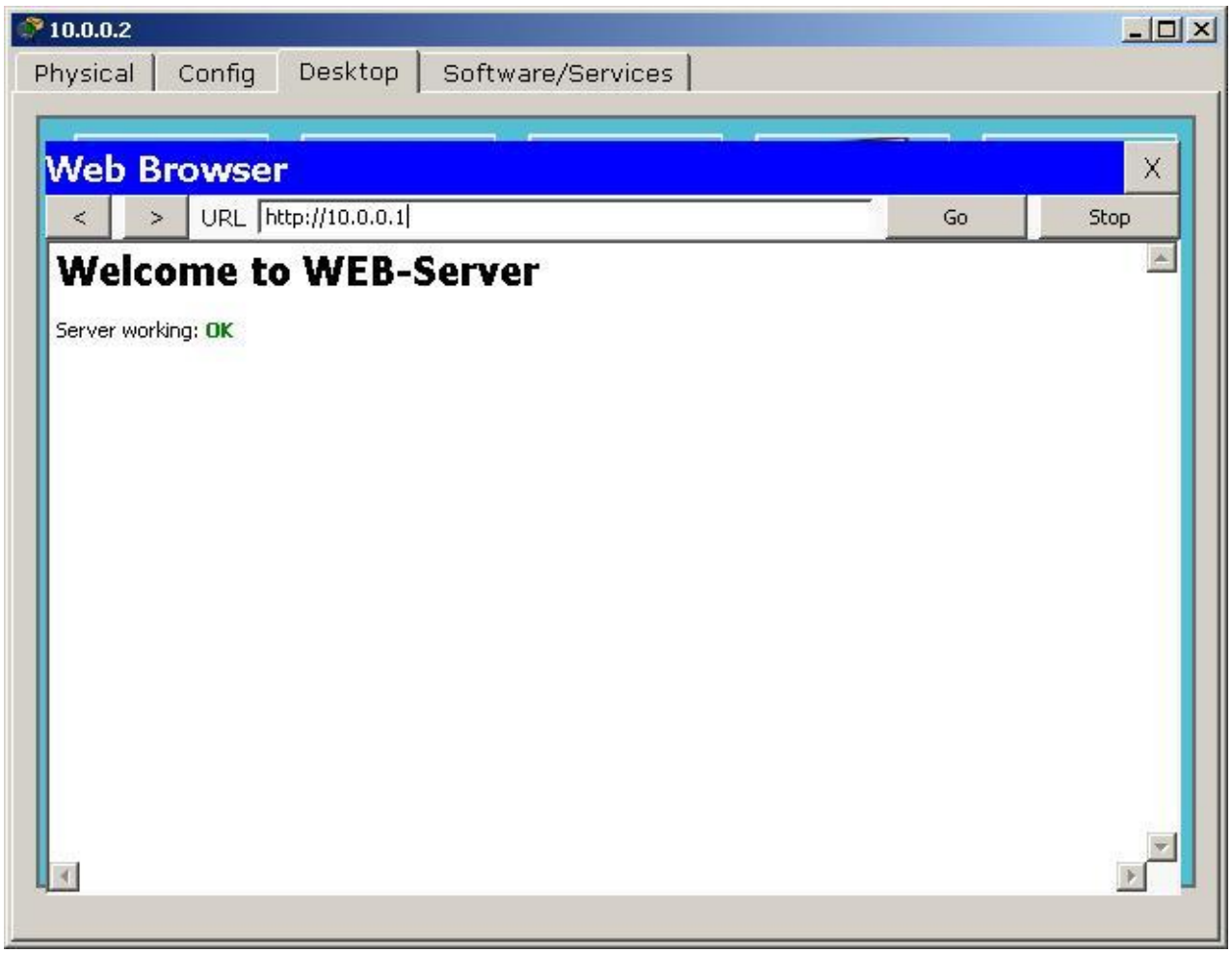


Рис.П4.4 Сервер работает

Приложение 5

Организация парольной защиты Cisco устройств

Организация парольной защиты в Cisco устройстве на примере коммутатора c2960 и маршрутизатора 1841. Прежде чем начать рассказ об организации парольной защиты и разграничении доступа в устройствах Cisco, позвольте представить вам топологию:

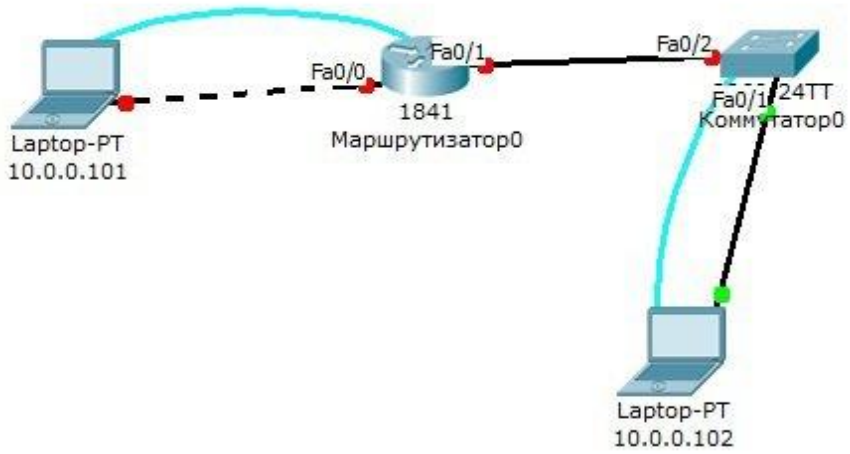


Рис. П5.1 Топология примера

Итак, привожу список задач, которые мы будем решать в топологии:

1. Устанавливаем пароль **enable**-режима на устройствах Маршрутизатор и Коммутатор равными соответственно *router* и *switch*;
2. Создаём пользователя **admin** с паролем cisco, который может логиниться по **telnet**-у;
3. Шифруем пароли, чтобы они не показывались в show running-config;
4. Назначаем IP адреса на устройства (10.0.0.1 и 10.0.0.2 соответственно), чтобы можно было подключаться удалённо;

Пока хватит, остальное выходит за рамки статьи. Итак, первым делом подключаемся через терминал с ноутбуков на устройства:

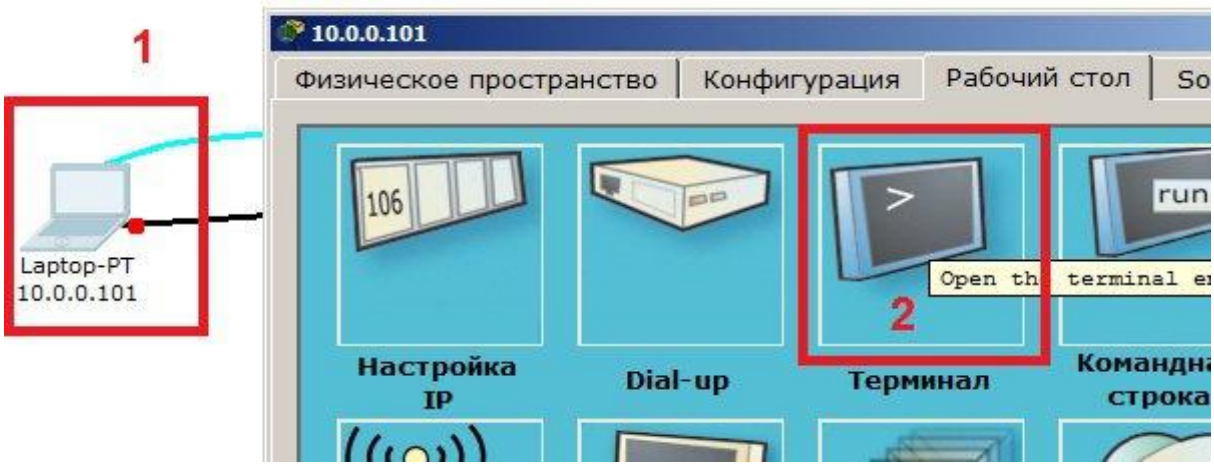


Рис.П5.2 Подключение с ноутбука по терминалу

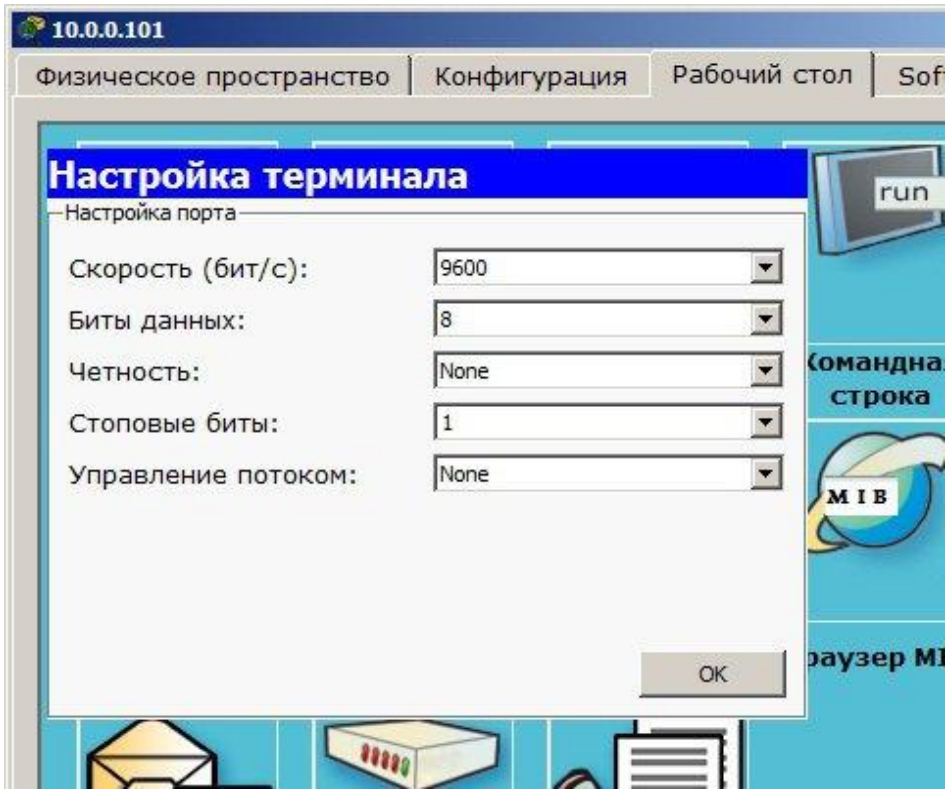


Рис.П5.3 Окно настройки терминала

```
Continue with configuration dialog? [yes/no]: no
Press RETURN to get started!
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

Таким образом мы перешли в режим глобального конфигурирования. Отсюда задаём пароль на enable.

```
Router(config)#enable secret router
```

Аналогичные действия на коммутаторе:

```
Switch(config)#enable secret switch
```

Проверяем:

Для этого выходим в приглашение пользователя (exit, exit, exit):

```
Router>en

Password: router

Router#
```

Отлично, с этим покончили! Следующий пункт, создать пользователя:

```
Router(config)#username admin privilege 0 secret cisco
```

0 — уровень привилегий. Может меняться от 0 до 15, 0 — самый низкий, далее — в порядке возрастания. Давать рекомендую всегда минимальный набор прав.

```
Router(config)#line vty 0 15
Router(config-line)#login local
```

Таким образом мы взяли все виртуальные терминалы (подключаемые, так как есть ещё con — консольный) и применили к ним local-авторизацию, то есть ту, в которой используются внутренние пользователи. В ином случае нужно было бы опустить слово local, но потребовалось бы указать пароль.

Прежде, чем проверить — нужно подключиться. Но подключиться мы не можем, пока не дадим IP адрес на устройство.

Дать IP адрес на роутер:

```
Router(config)#interface fastEthernet0/0
Router(config-if)#ip address 10.0.0.1 255.255.255.0
Router(config-if)#no shutdown
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
Router(config-if)#exit
```

Дать IP адрес на коммутатор:

```
Switch(config)#interface vlan 1
Switch(config-if)#ip address 10.0.0.2 255.255.255.0
Switch(config-if)#no shutdown
%LINK-5-CHANGED: Interface Vlan1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to
up
Switch(config-if)#exit
```

Проверяем коннект:

Входим на ноутбук, подключенный к роутеру и ищем рядом с терминалом в программах командную строку. Посылаем Echo-запрос на нужный IP-адрес:

```
PC>ping 10.0.0.1
Pinging 10.0.0.1 with 32 bytes of data:
Reply from 10.0.0.1: bytes=32 time=65ms TTL=255
Reply from 10.0.0.1: bytes=32 time=4ms TTL=255
Reply from 10.0.0.1: bytes=32 time=4ms TTL=255
Reply from 10.0.0.1: bytes=32 time=2ms TTL=255
```

Как видим, роутер отвечает. Пробуем подключиться:

```
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open
```

```
User Access Verification
Username: admin
Password: cisco
Router>en
Password: router
Router#
```

Шифруем пароли.

Вы можете убедиться, что по команде:

```
Router# show running-config
```

выводится весь конфиг и пароли там в открытом виде. Для того, чтобы это избежать — используем операцию:

```
Router(config)#service password-encryption
```

Теперь, при прочтении конфига они выглядят так:

```
Building configuration...
Current configuration : 601 bytes
!
version 12.4
no service timestamps log datetime msec
no service timestamps debug datetime msec
service password-encryption
!
hostname Router
!
enable secret 5 $1$mERr$uP0U5aamVaETEvWzvDbvp.
!
username admin privilege 0 secret 5 $1$mERr$hx5rVt7rPNoS4wqbXKX7m0
```

Ограничить доступ узлам в VLAN

Раз возникла у меня задача. В общей сети нужно было задать два узла (на разных коммутаторах), которые могли бы обмениваться данными между собой, но используя общую сеть. Так, чтобы никто другой не мог получить к ним доступ. И они при всём желании не могли бы никаким образом взаимодействовать с узлами в моей сети. Одним словом, выделить их в отдельную подсеть и запереть там навсегда.

Для начала я выбрал в качестве решения сеть с VLAN-ами, по стандарту IEEE 802.1Q, когда можно маркировать пакеты определённым тегом и пакет может быть доступен только на портах, отмеченных нужным тегом.

Решено было промаркировать два порта на соответствующих свичах VLAN-ом, например с номером «10», и выделить им отдельные адреса. Даже если бы они взяли адреса из моей сети – то всё равно не могли бы получить доступ к общим ресурсам, так как на порту жестко привязан VLAN.

Смотрим схему:

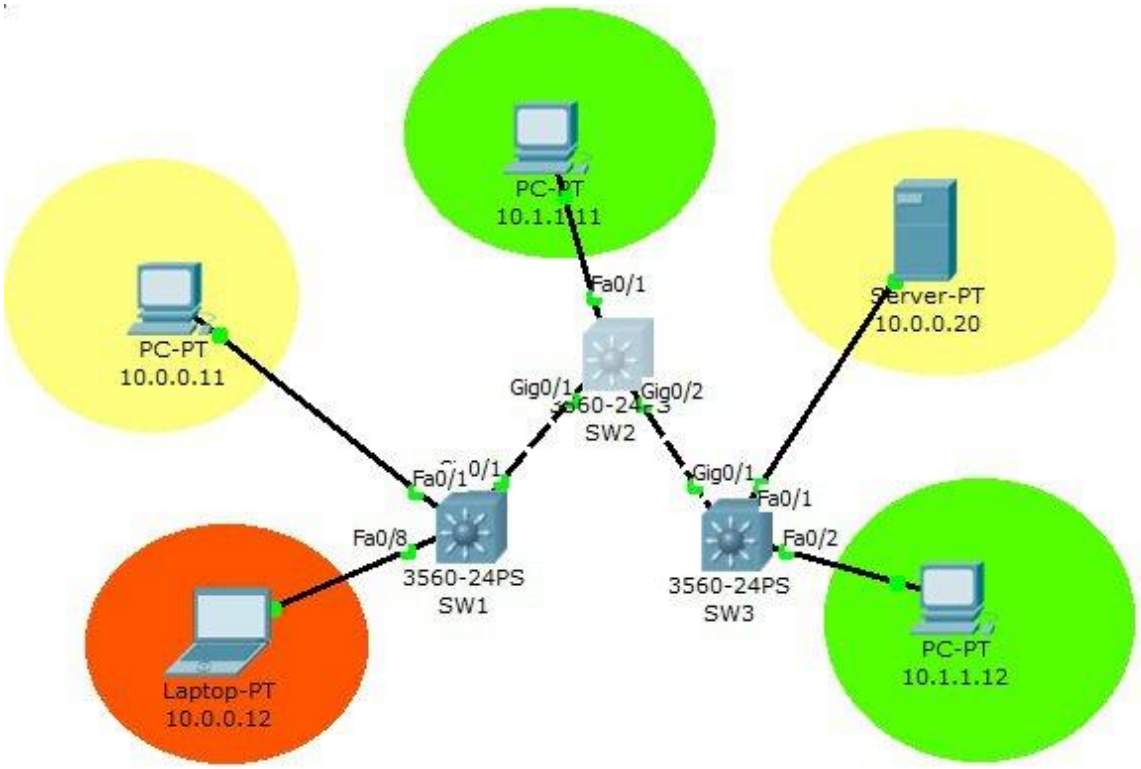


Рис. П6.1 Пример сети

Зелёными областями я выделил свою сеть, которая является основной на предприятии. Желтыми областями – два узла, которые нужно соединить, красным – ноутбук потенциального нарушителя, взявшего себе адрес из «выделенной» подсети и пытающегося получить доступ к этим узлам.

Так же тут имеются три коммутатора 3-его уровня (Cisco 3560), которые соединены кроссовер-кабелем. Номера портов тут подписаны, так что разобраться будет не сложно.

Первым делом настраиваем коммутаторы. Нужно соединить их транк-портами. Напомню, что trunk – пропускает трафик без учёта VLAN-ов, то есть тегированный и нетегированный.

```
SW1(config)#int gi0/1
SW1(config-if)#switchport trunk encapsulation dot1q
SW1(config-if)#switchport mode trunk
```

То же самое делаем со всеми портами, соединяющими коммутаторы. С обеих сторон.

Кстати, переходим в режим симуляции и пропингуем желтым хостом желтый сервер (для простоты буду давать цвета).

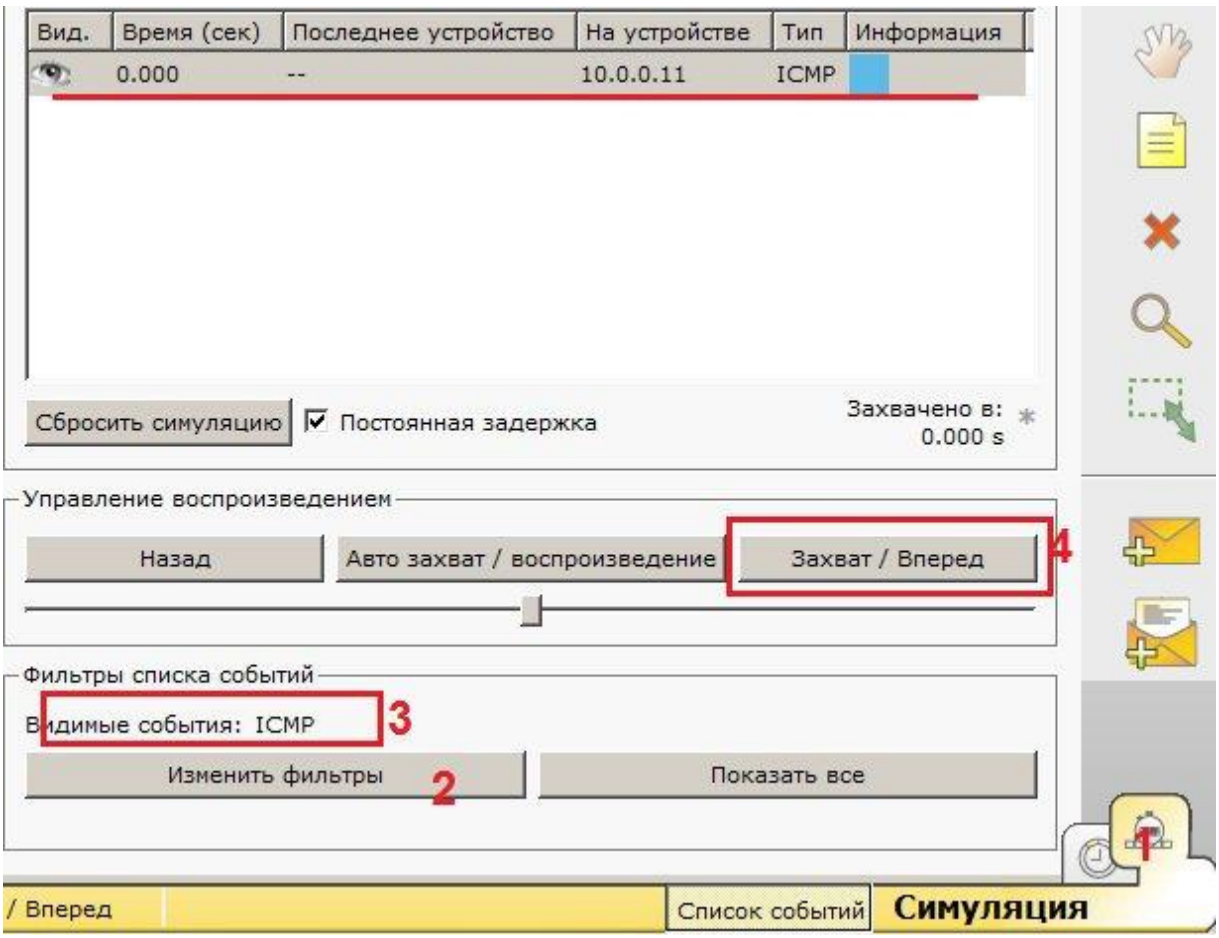


Рис.П6.2 Процесс симуляции прохождения пакетов ping

Выставили фильтры в режими симуляции (это переключатель справа «1»), выбираем фильтрами протокол ICMP «2», «3» и посылаем ping-запрос. Вот, что находится в отловленном пакете (конверт):

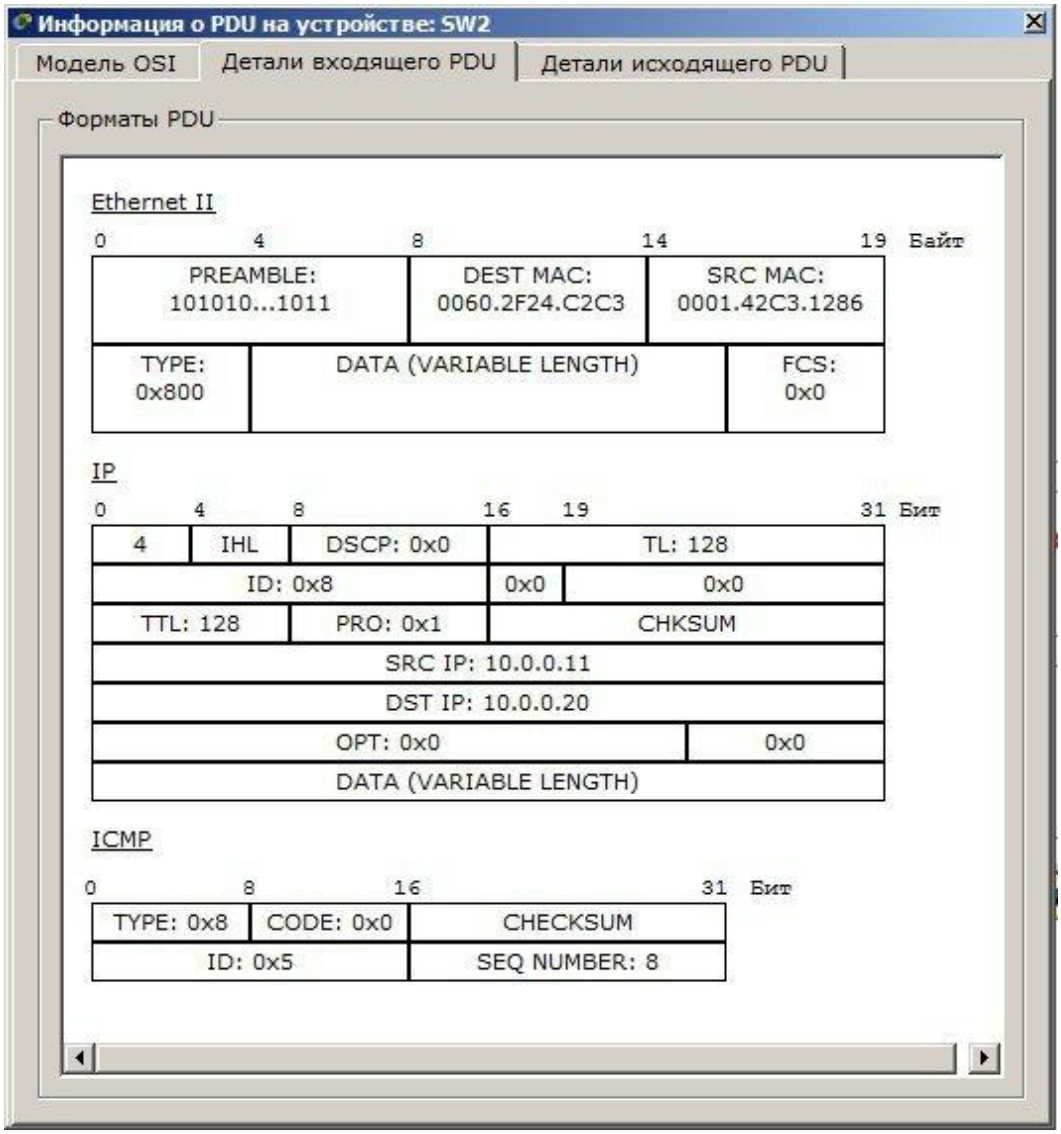


Рис.П6.3 Детальные данные фрейма Ethernet

Обращаем внимание на 2-ой уровень (Ethernet-фрейм). Обычный кадр, ничем не примечателен. А теперь пометим соответствующие желтым хостам порты нужным нам VLAN-ом (10-ым, если вы помните).

```
SW1(config)#int fa0/1
SW1(config-if)#switchport mode access
SW1(config-if)#switchport access vlan 10
SW3(config)#int fa0/1
SW3(config-if)#switchport mode access
SW3(config-if)#switchport access vlan 10
```

Повторим пинг. Увидим уже в отловленном пакете **признаки инкапсуляции в 802.1q** фрейм:

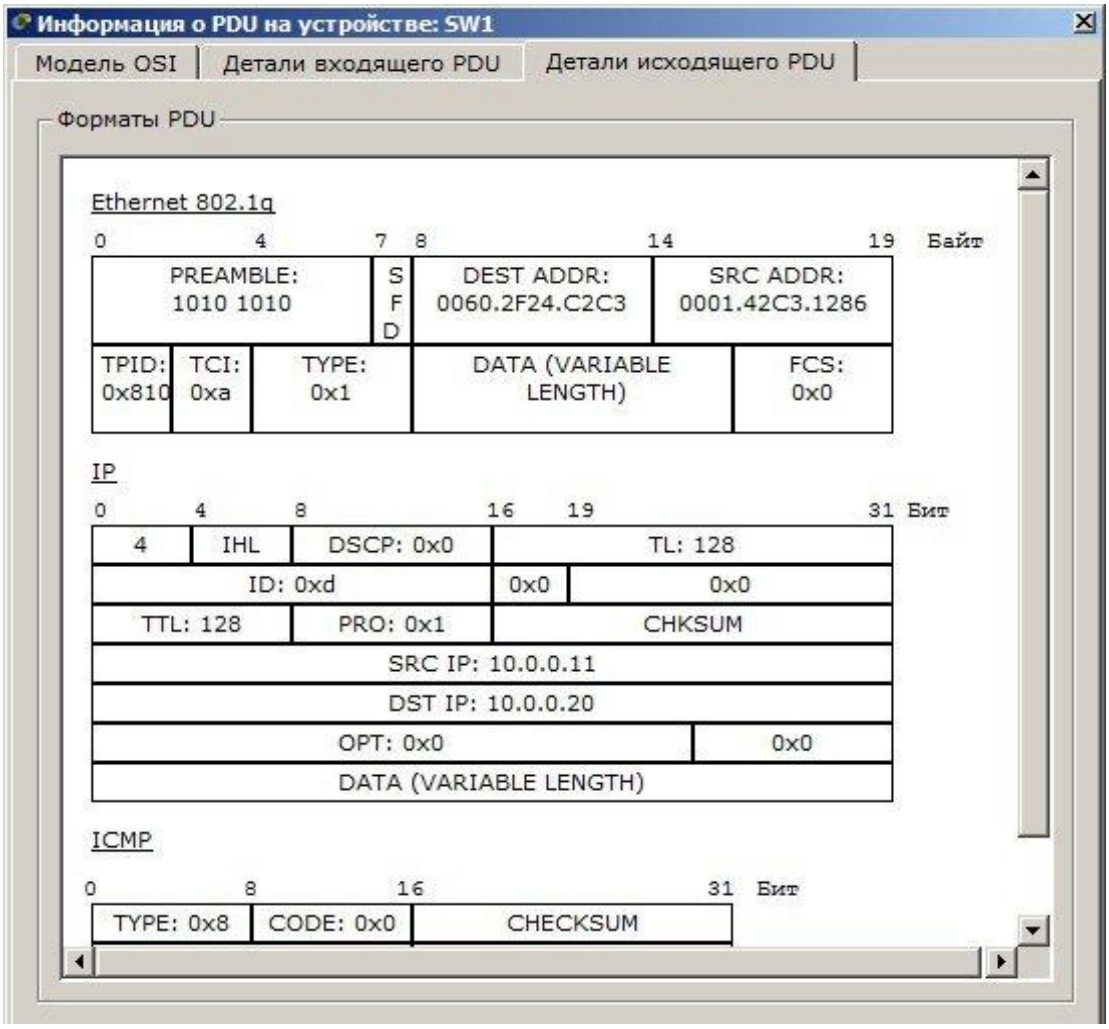


Рис.П6.4 Детальные данные фрейма Ethernet с информацией о VLAN

И наш номер VLAN-а в поле TCI:0xa (hex:0xA =10). Вот трафик и помечен. Через транковые порты он проходит без проблем. А вот на всех остальных он отбрасывается, если не задан mode access с нужным VLAN-ом.

Даже если злоумышленник сменит IP, выбраться за пределы 10-го VLAN-а ему не светит. А раз доступа к настойке коммутаторов у него потенциально нет, то, можно сказать, что задача решена.

Приложение 7

Протокол DHCP

С помощью *Протокола динамической конфигурации узла* компьютер, подключенный к сети, в котором есть DHCP-сервер, может беспрепятственно получить сетевой адрес, маску сети, адрес шлюза, DNS и прочую сетевую информацию. Для тех, кто не в курсе, существует еще другой вид сетевой идентификации нового узла — всего-навсего ввести все вручную. Такое явление называется статическим режимом. На то оно и название — динамическая (т.е. автоматическая) настройка сети. А бывает, как вы догадались, статическая (т.е. ручная) настройка.

Суть работы и практическое применение DHCP

Представьте себе следующую ситуацию. Есть небольшой офис, который с каждым месяцем становится все больше и больше. Больше в том плане, что постоянно докупаются новые компьютеры и «хабы». Администратор или технический специалист (в подчинении у администратора) настраивает помимо системы, ярлычков, деинсталляции Angry Birds (:-)). Теперь еще и параметры сети. Ведь надо, чтобы на рабочем месте работали все сетевые папки и Интернет. Для этого на каждом компьютере надо зайти в свойства сетевого соединения и прописать новый IP-адрес, маску, шлюз и т.д. В небольшом офисе, где 7 машин - такой подход ещё приемлем.

Но когда становится больше десятка машин, скажем, 20 машин, что тогда делать? Можно и запутаться, какой адрес у нас есть в сети, а какого нет. Вот пример. Хорошо, кажется вспомнили, что адреса 192.168.1.9 у нас еще не было. Ввели, ошибок не выдало. Оставляем работать. Но позже оказывается, что компьютер с таким адресом есть в сети, а приняла сеть такой адрес потому, что в тот момент тот компьютер был выключен. Очень неприятная ситуация, особенно когда настройщик сети находитесь далеко. А объяснять тетеньке, которой осталось 3 года до пенсии, что надо исправить

— очень тяжело. Вот возможно, поэтому и придумали автоматическую настройку. Как она работает?

После подключения к сети клиент, настроенный на использование DHCP, производит по протоколу UDP широковещательную рассылку. Это пакет на адрес 255.255.255.255 по порту 68. Цель рассылки — обнаружить DHCP-сервер. Сервер обнаружен, он и назначает нашему клиенту свободный адрес (по порту 67). Кроме адреса сервер передает еще и другие сетевые параметры, заданные администратором (IP-адрес DNS, шлюз и т.д.).

Пример взаимодействия

Клиент посылает пакет на широковещательный адрес (достигает нашего сервера). Кстати, настройка DHCP на оборудовании Cisco.

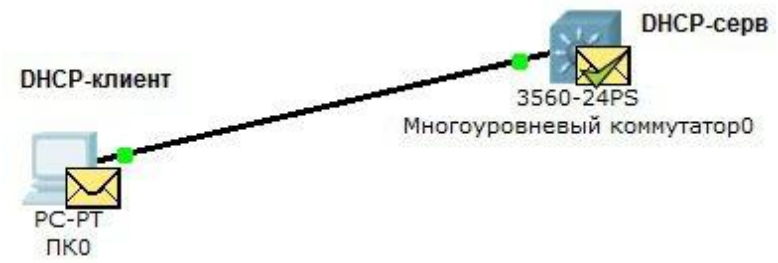


Рис.П7.1 Взаимодействие клиент-сервер DHCP

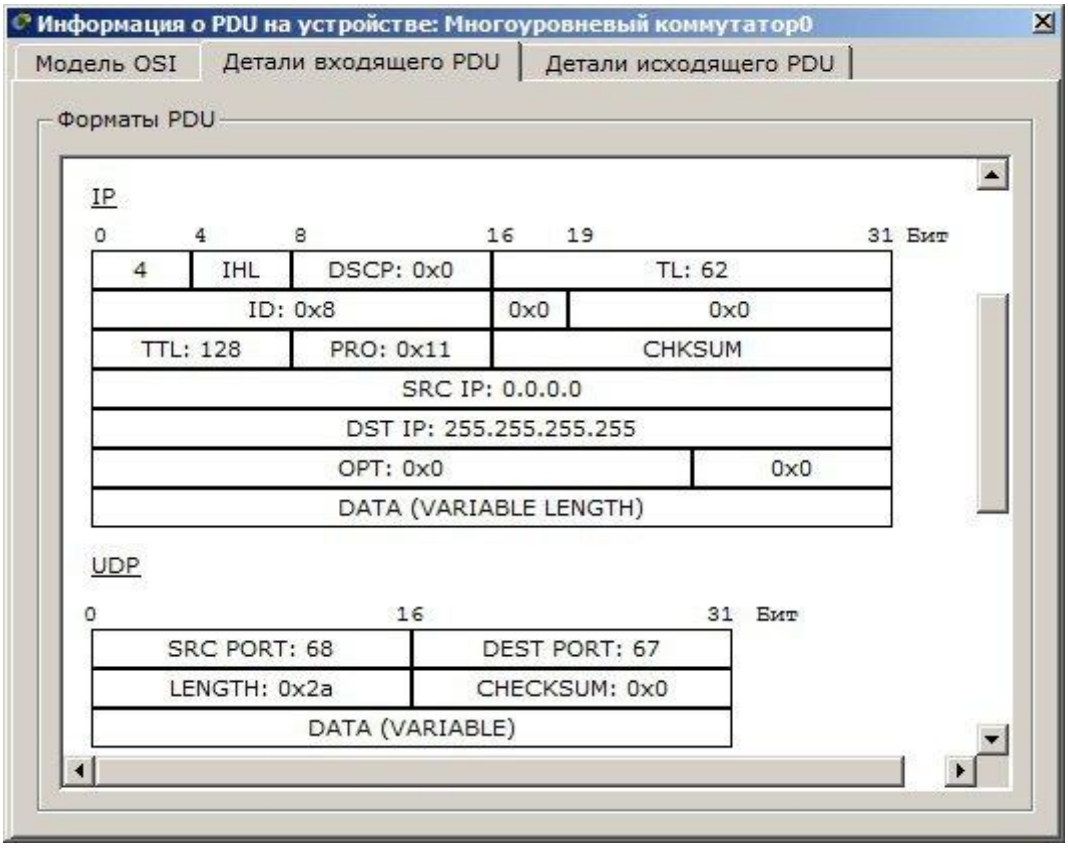


Рис.П7.2 Детализация пришедшего пакета:

Обратите внимание на детали пакета IP:

- SRC IP: 0.0.0.0 — текущий.
- DST IP: 255.255.255.255 — широковещательный адрес (в пределах одного домена).

Детали пакета UDP:

- SRC PORT: 68
- DEST PORT: 67

В ответ возвращается пакет с настройками сети для клиента:

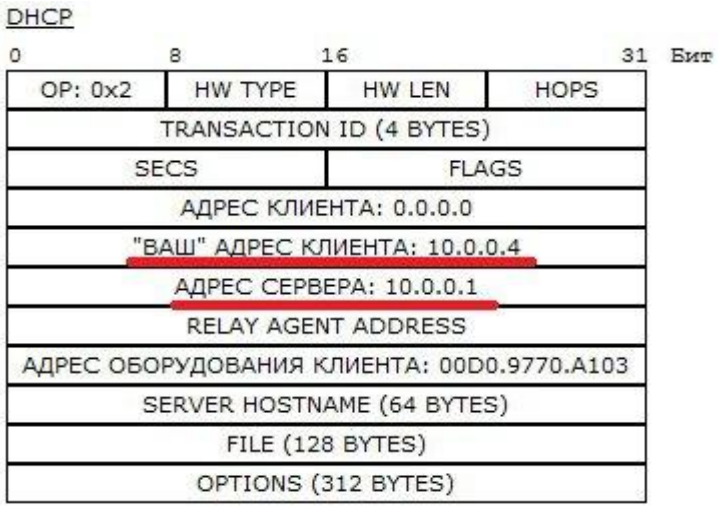


Рис. П7.3 Пакет ответа Сервера

Несколько нюансов по DHCP

- Адрес дается не навсегда, а на время аренды. Аренда задается тоже на сервере.
- Сервер можно настроить так, что определенному клиенту он будет назначать фиксированный адрес. Это бывает очень удобно. Достигается это путем запоминания уникального адреса сетевого устройства — MAC-адреса.
- DHCP-сервером может являться не только сам компьютер-сервер, но и другое устройство. Наиболее распространенный случай — роутер (или рутер, или маршрутизатор) со встроенным DHCP. Очень удобно и дешево.

Понятно, что такая процедура технически сложнее, чем статический режим. Для грамотной настройки требуются небольшие умения и знания по администрированию. Но иначе зачем читать эту статью! Если потратить полчаса на настройку сервера, можно будет централизованно управлять конфигурацией сети. Тогда стоит только изменить IP-адрес DNS-сервера в конфигурации DHCP — на остальных компьютерах сети новые IP-адреса DNS пропишутся автоматически.

Настраиваем DHCP-службу на оборудовании Cisco

Нарисуем простую топологию:

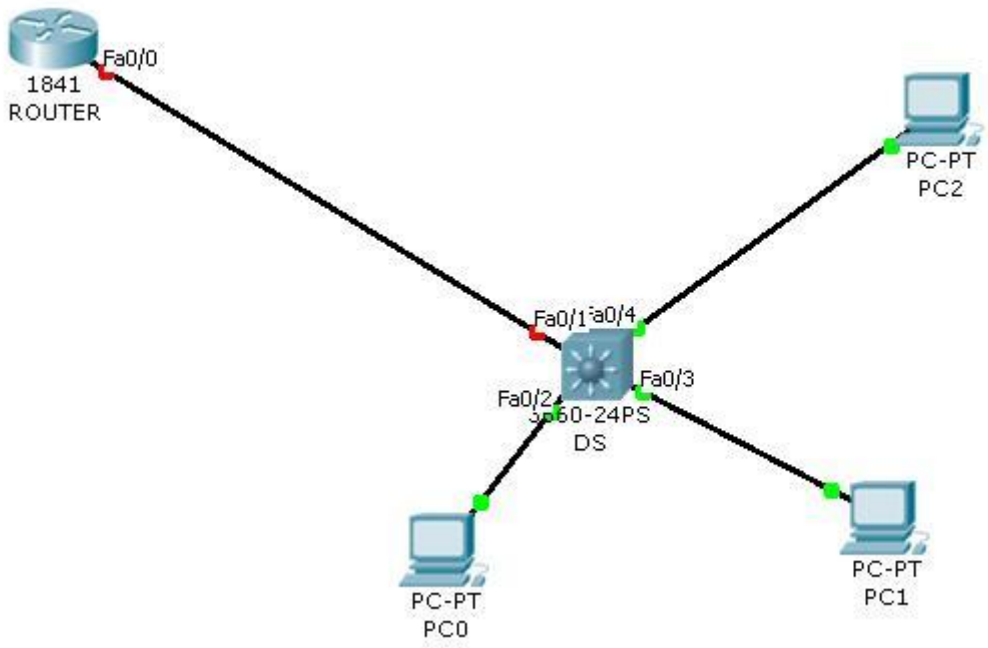


Рис.П7.4 Пример топологии

Настраиваем интерфейс роутера

```
r01(config)#int fa0/0
r01(config-if)#ip address 10.0.0.1 255.255.255.0
r01(config-if)#no shut
```

Мы просто назначили адрес из интересующей нас подсети на интерфейс. Как только на свиче порт поднимется, будем настраивать DHCP-сервер.

Исключаем адрес самого порта из списка раздаваемых. Вообще, этой командой можно задать адреса, которые чем-то зарезервированы. Например DNS или сервера.

```
r01(config)#ip dhcp excluded-address 10.0.0.1
```

Настраиваем пул адресов:

```
r01(config)#ip dhcp pool myDHCP
r01(dhcp-config)#default-router 10.0.0.1
r01(dhcp-config)#dns-server 10.0.0.254
r01(dhcp-config)#network 10.0.0.0 255.255.255.0
r01(dhcp-config)#exit
r01(config)#
```

Создали пул DHCP-адресов, шлюз по-умолчанию 10.0.0.1, ДНС соответственно .254, маска подсети /24.

Проверяем получение адресов:

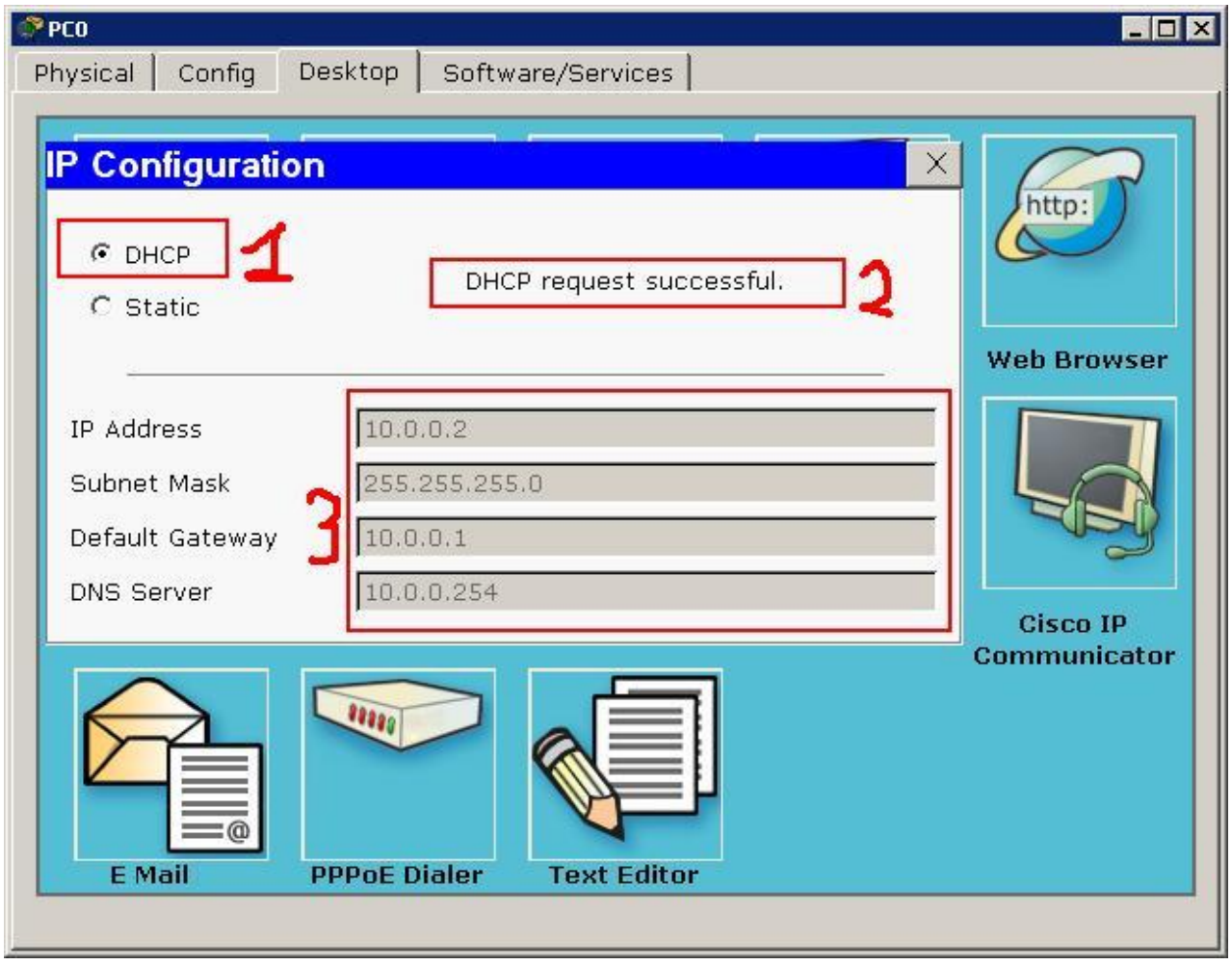


Рис.П7.5 Проверка получения сетевых настроек на компьютере 1

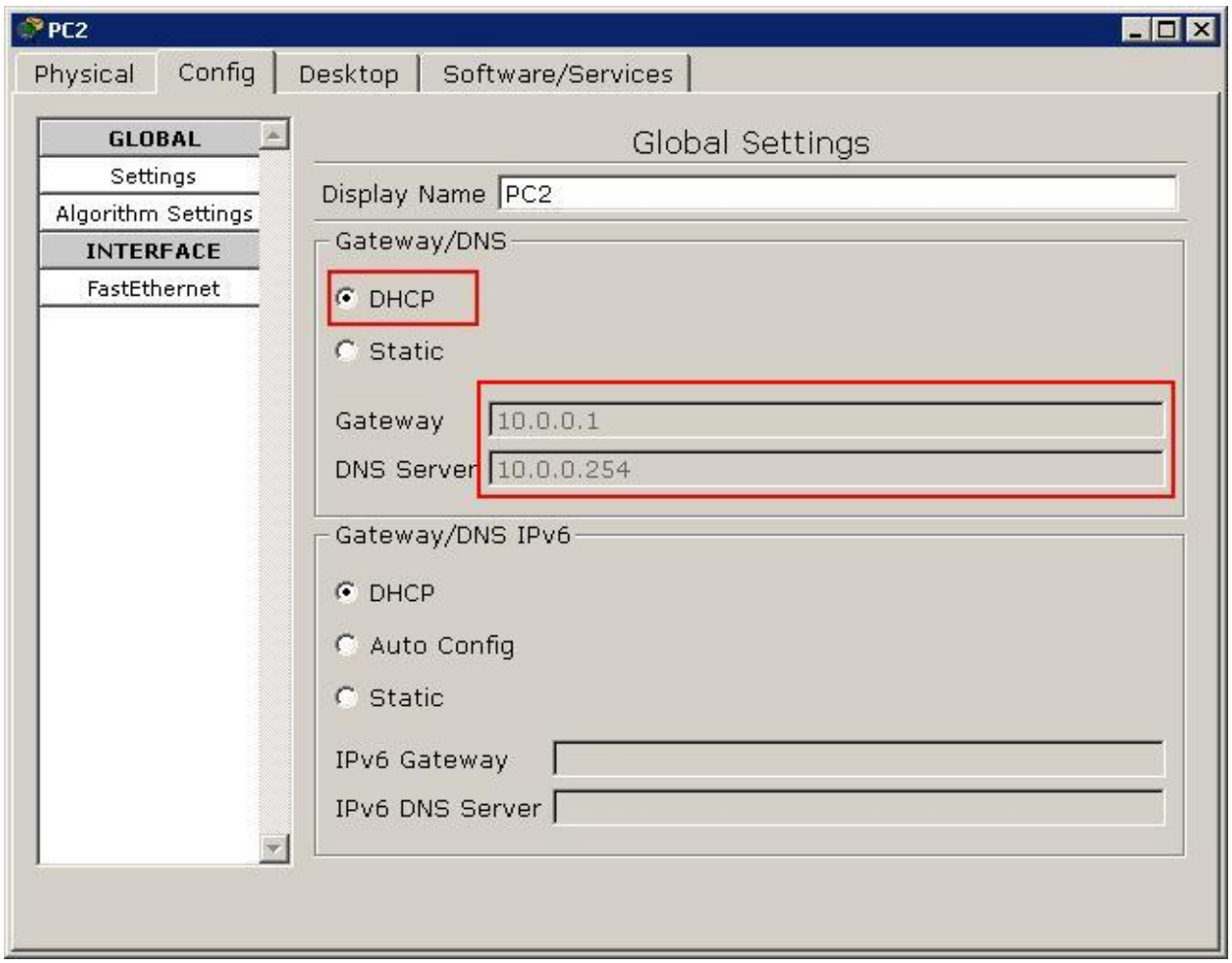


Рис.П7.6 Проверка получения сетевых настроек на компьютере 2

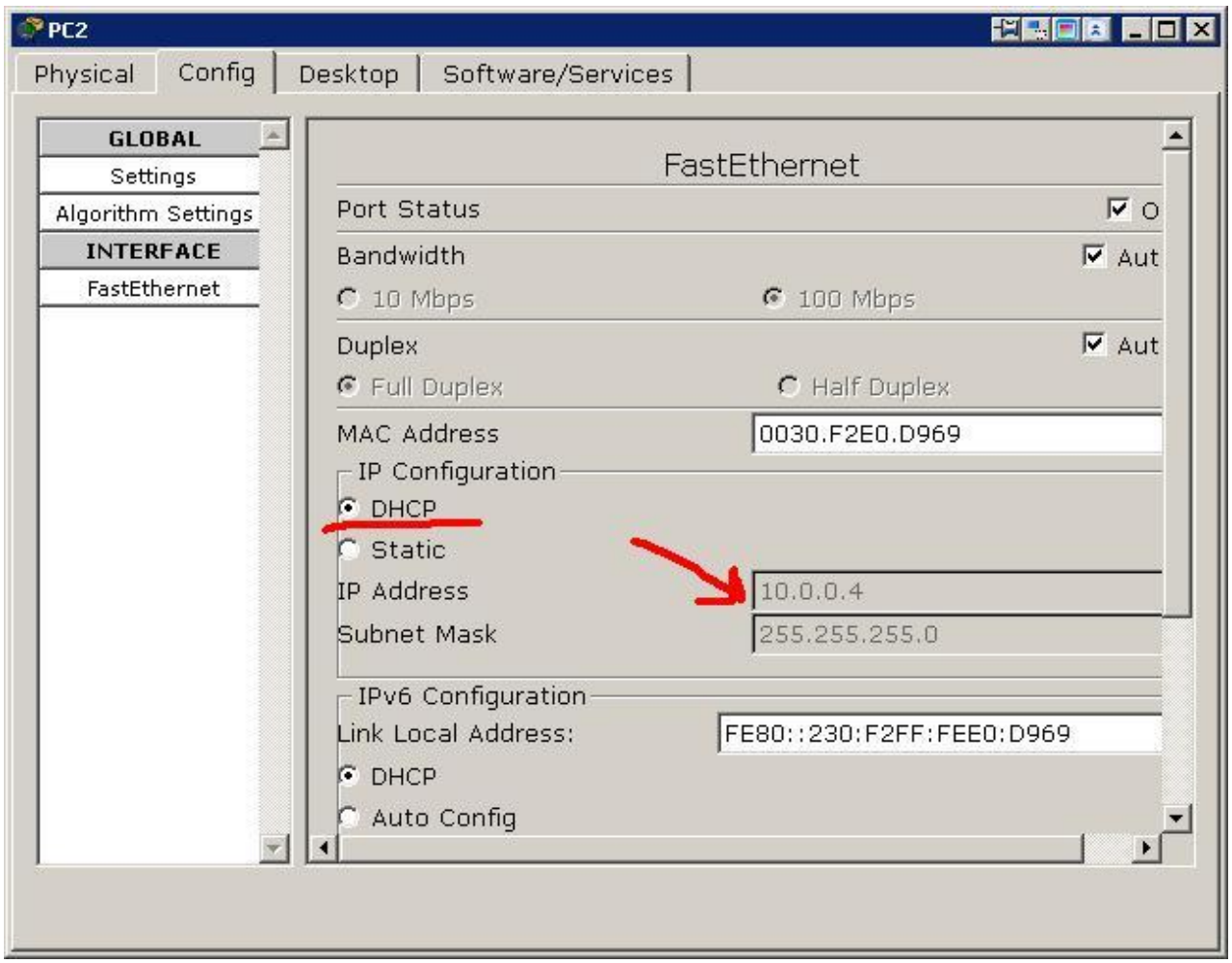


Рис.П7.7 Проверка получения сетевых настроек на компьютере 3

А теперь немного изменим задачу. Раздавать адреса будем и с коммутатора (уберём роутер нафик).

Новая топология.

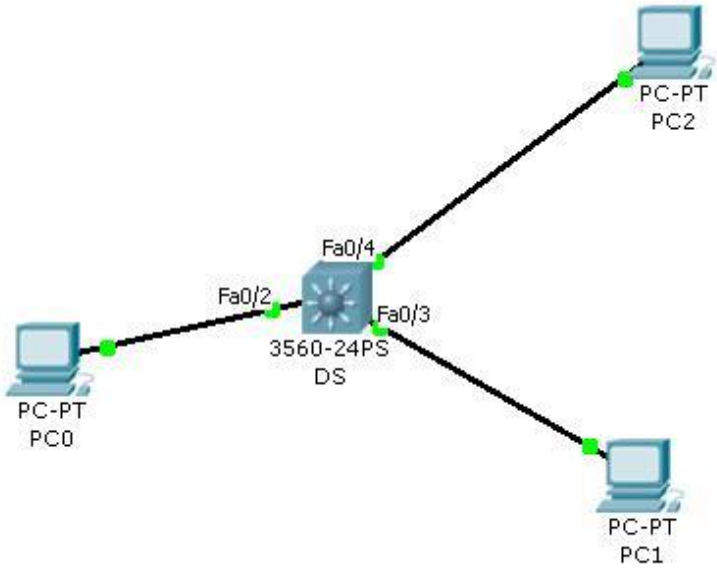


Рис.П7.8 Измененная топология

```
SW>en
SW#conf t
Enter configuration commands, one per line. End with CNTL/Z.
```

Назначаем адрес на VLAN 1

```
SW(config)#int vlan 1
SW(config-if)#ip address 192.168.100.1 255.255.255.0
SW(config-if)#no shut
SW(config-if)#exit
```

Если планируется настроить другой VLAN — выбираем его, также смотрите, чтобы порты, к которым подключено оборудование принадлежащее нужному VLAN.

Создаём пул адресов аналогично предыдущему примеру:

```
SW(config)#ip dhcp excluded-address 192.168.100.1
SW(config)#ip dhcp pool newMy
SW(dhcp-config)#default-router 192.168.100.1
```

```
SW(dhcp-config)#dns-server 192.168.100.254
SW(dhcp-config)#network 192.168.100.0 255.255.255.0
SW(dhcp-config)#exit
SW(config)#
```

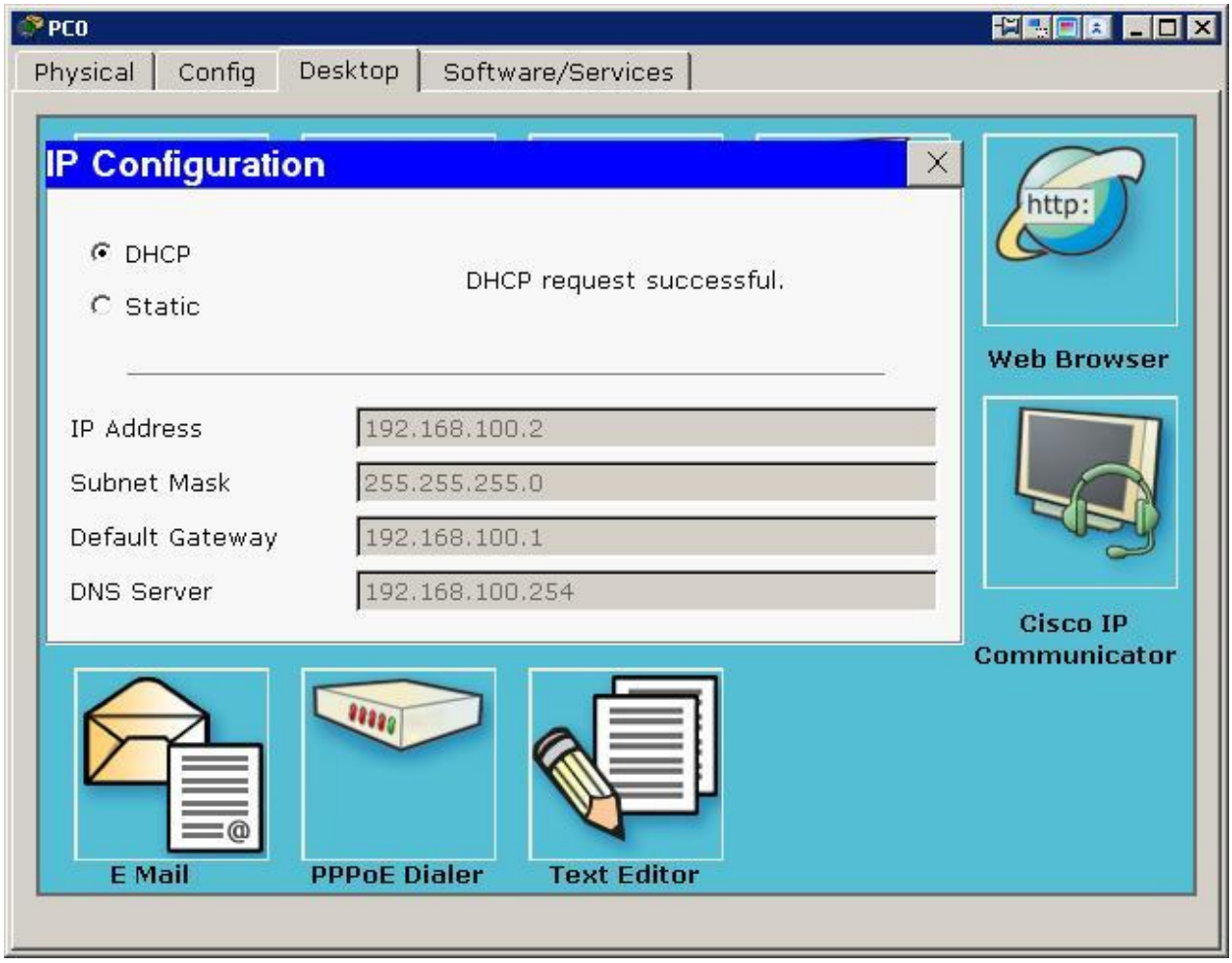


Рис.П7.9 Проверка получения сетевых настроек на компьютере