Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Томский государственный университет систем управления и радиоэлектроники»

Кафедра телекоммуникаций и основ радиотехники

СЕТИ ЭВМ И ТЕЛЕКОММУНИКАЦИИ

**Методические указания
по проведению практических занятий и
организации самостоятельной работы студентов,
обучающихся по направлению подготовки магистров 210700
«Инфокоммуникационные технологии и системы связи»**

2012

Богомолов, Сергей Ильич

Сети ЭВМ и телекоммуникации: Методические указания по проведению практических занятий и организации самостоятельной работы студентов, обучающихся по направлению подготовки магистров 210700 «Инфокоммуникационные технологии и системы связи». – Томск: Томский государственный университет систем управления и радиоэлектроники, 2012. — 70 с.

Учебно-методическое пособие предназначено для студентов дневной формы обучения, обучающихся по программе магистерской подготовки. Цель пособия – оказать помощь преподавателям и студентам в вопросах проведения практических занятий и организации самостоятельной работы при изучении дисциплины «Сети ЭВМ и телекоммуникации».

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Томский государственный университет систем управления и
радиоэлектроники»

Кафедра телекоммуникаций и основ радиотехники

УТВЕРЖДАЮ
Зав. кафедрой ТОР
_____Е.П. Ворошилин
«___» _____ 2012 г.

СЕТИ ЭВМ И ТЕЛЕКОММУНИКАЦИИ

Методические указания
по проведению практических занятий и
организации самостоятельной работы студентов,
обучающихся по направлению подготовки магистров 210700
«Инфокоммуникационные технологии и системы связи»

Разработал:
Доцент каф. ТОР
_____ С.И. Богомолов
_____ 2012 г.

2012

СОДЕРЖАНИЕ

1 ВВЕДЕНИЕ

Целью изучения дисциплины «Сети электронно-вычислительных машин и телекоммуникации» является приобретение студентами знаний о принципах построения современных локальных и глобальных сетей ЭВМ, функционировании уровней модели OSI при взаимодействии прикладных процессов, базовых технологиях локальных сетей, стандартных стеках протоколов, принципах маршрутизации, аппаратных и программных средствах телекоммуникаций, приобретение знаний и навыков, необходимых для профессиональной деятельности.

Практические занятия имеют целью закрепление навыков и умений практической работы в компьютерной сети, обоснования и контроля основных сетевых параметров, первичных навыков администрирования сети, а также работы в режиме удаленного доступа. В ходе выполнения лабораторных работ студенты изучают процессы взаимодействия сетевых компонентов путем симуляции функционирования различных фрагментов сети, а также исследуют основные характеристики сетевых протоколов.

Самостоятельная работа студентов по дисциплине «Сети ЭВМ и телекоммуникации» содержит следующие основные составляющие: проработка лекционного материала, подготовка к практическим занятиям и лабораторным работам и выполнение отчетов, изучение вопросов лекционного курса, вынесенных на самостоятельное изучение.

Проработка лекционного материала не требует особых методических указаний. Рекомендуется просматривать материалы лекции в тот же день после ее окончания, как говорится, «по горячим следам». В данном пособии рассмотрены вопросы организации самостоятельной работы при подготовке к практическим занятиям и лабораторным работам и при изучении тем лекционного курса, вынесенных на самостоятельное изучение.

В качестве основного источника изучения по данной дисциплине следует использовать учебное пособие [1.1]. Кроме того, могут быть использованы разнообразные дополнительные материалы, в том числе и приведенные в списке рекомендуемой литературы [2.1-2.4]. Этот список литературы может быть рекомендован также и при подготовке к практическим занятиям и лабораторным работам, а также при изучении тем лекционного курса, вынесенных на самостоятельное изучение.

При выполнении лабораторных работ используется свободно распространяемое программное обеспечение: пакеты программ сетевых симуляторов NS2 и Net-Simulator а также программы и утилиты, входящие в состав операционной системы Linux.

Программное и методическое обеспечение, используемое при изучении данной дисциплины, размещено на сайтах соответствующих разработчиков этих продуктов и доступно для свободного использования. Копии этих продуктов продублированы на сервере локальной сети кафедры ТОР. Перечень этих материалов приведен в списке используемой литературы.

2 СОДЕРЖАНИЕ РАЗДЕЛОВ ДИСЦИПЛИНЫ

2.1 Содержание лекционного курса

Введение

Основные понятия и определения. Эволюция вычислительных систем. Многопроцессорные системы. Многомашинные системы. Вычислительные сети.

Основные характеристики вычислительных сетей

Задачи и проблемы распределенной обработки данных. Механизмы взаимодействия сетевых компонентов. Топология сети. Сетевые технологии. Сетевые службы. Структуризация сетей.

Понятие «открытая система» и проблемы стандартизации. Уровни, протоколы и интерфейсы. Иерархия протоколов. Модель OSI. Основные сетевые стандарты.

Классификация сетей по способам распределения данных, сравнительная характеристика различных типов сетей. Основы организации и функционирования сетей. Средства взаимодействия процессов в сетях. Распределенная обработка информации в системах клиент-сервер. Одноранговые сети. Неоднородные вычислительные сети.

Основы телекоммуникации

Линии связи. Сигналы линий связи. Модуляция сигналов. Цифровое и логическое кодирование. Компрессия данных.

Синхронные и асинхронные протоколы канального уровня. Передача с установлением соединения и без установления соединения. Обнаружение и коррекция ошибок.

Методы коммутации. Коммутация каналов. Коммутация пакетов. Коммутация сообщений.

Локальные сети

Протоколы и стандарты локальных сетей.

Технология Ethernet. Основные характеристики технологии. Метод доступа CSMA/CD. Форматы кадров технологии Ethernet. Спецификации физической среды Ethernet.

Технологии стандартов IEEE 802.x. Основные характеристики. Форматы кадров канального уровня. Методы доступа к среде. Спецификации физического уровня.

Оборудование локальных сетей. Сетевые адаптеры. Концентраторы. Коммутаторы.

Объединение сетей

Интеграция локальных сетей в региональные и глобальные сети. Принципы маршрутизации. Реализация межсетевого взаимодействия средствами TCP/IP.

Адресация в IP сетях. Типы адресов стека TCP/IP: локальные адреса, сетевые адреса, доменные имена. Использование масок в IP адресации. Взаимное отображение имен и адресов.

Основные функции протокола IP. Структура IP пакета. Маршрутизация в IP сетях. Фрагментация IP пакетов. Протокол надежной доставки TCP сообщений. Протоколы маршрутизации в IP сетях.

Средства построения составных сетей стека Novel.

Глобальные сети

Основные понятия и определения. Глобальные сети на основе выделенных линий. Глобальные сети на основе сетей с коммутацией каналов. Глобальные сети на основе сетей с коммутацией пакетов.

Internet, основные службы и предоставляемые услуги, стандарты, перспективы развития. Организация корпоративных сетей.

Протоколы электронной почты и телеконференций Internet. Принципы создания Web-узла.

Сетевые операционные системы (ОС)

Назначение и функции операционной системы. Архитектура операционной системы.

Сетевые средства UNIX: основные протоколы, службы, функционирование, сопровождение и разработка приложений, особенности реализации на различных платформах. Сетевая операционная система Windows NT основные протоколы, службы, функционирование, генерация, сопровождение и разработка приложений. Сетевая операционная система Novel NetWare: основные протоколы, службы, функционирование, генерация, сопровождение и разработка приложений.

Эффективность функционирования вычислительных сетей и перспективы их развития

Основные понятия безопасности. Средства идентификации и аутентификации.

Показатели эффективности вычислительных сетей. Средства повышения надежности функционирования сетей.

Перспективы развития вычислительных сетей и телекоммуникаций.

2.2 Перечень лабораторных работ

Исследования основных компонентов сетевого имитатора NS2.

Моделирование сетей ЭВМ с помощью сетевого имитатора NS2.

Исследование характеристик протокола TCP с помощью сетевого имитатора NS2

2.3 Темы практических занятий

Локальные сети. Сети Ethernet.

Сетевой уровень. Адресация в IP сетях. Таблицы маршрутизации.

Маршрутизация в IP-сетях. Использование масок.

Элементы диагностики сети.

Удаленный доступ. Безопасность работы в сети.

Исследование сетевых компонентов с помощью имитатора Net-simulator.

3 ПРАКТИЧЕСКИЕ ЗАНЯТИЯ

3.1 Локальные сети. Сети Ethernet.

При подготовке к данному занятию использовать соответствующие разделы лекций и материалов, указанных в перечне литературы. Особое внимание обратить на следующие вопросы: протоколы и стандарты локальных сетей; основные характеристики технологии Ethernet; метод доступа CSMA/CD; форматы кадров технологии Ethernet; спецификации физической среды Ethernet.

На практических занятиях выполняются расчеты характеристик сети Ethernet: расчет допустимого времени задержки распространения между двумя узлами и расчет производительности сети.

Расчет характеристик сети Ethernet.

Расчет PDV

При проектировании сетей Ethernet, реализуемых с помощью хабов, применяют две модели:

1. Первая модель применима для сетей, работающих с компонентами одного стандарта, например 10BaseT. Если выполняются требования, изложенные в спецификации, например, для 10BaseT (кабель - неэкранированная витая пара UTP; максимальная длина сегмента до 100 м; максимальное расстояние между узлами сети менее 500 м; максимальное число хабов 4), то сеть будет функционировать в штатном режиме (номинальная пропускная способность – 10 Мбит/с при максимальном число станций 1024).

В этом случае никаких расчетов проводить не надо. Часто эту модель называют как «правило четырех хабов».

2. Вторая модель основана на вычислении времени задержки для максимального расстояния между узлами. Эта методика, применима для неоднородных сетей, когда сегменты сети, выполнены и на ВОЛС и на витой паре, а также когда расстояния между хабами выходят за пределы допустимого. Расчету подлежит *PDV* – удвоенная задержка распространения

$t_s = L \cdot t_1 + t_0$,

где t_0 – задержка прохождения сигнала по цепям концентратора;

t_1 – задержка прохождения сигналом по линии длиной 1 метр;

L – расстояние между станциями.··

Необходимые данные для расчета задержки неоднородной сети (комбинация технологий 10BaseT и 10BaseFL) приведены в таблице.

Таблица 3.1. Задержки распространения сигнала

Тип сегмента	Макс. длина, м	Начальный сегмент		Промежуточный сегмент		Конечный сегмент		t_1
		t_0	t_m	t_0	t_m	t_0	t_m	
10BaseT	100	15,25	26,6	42,0	53,3	165	176,3	0,113
10BaseFL	2000	12,25	212,3	33,5	233,5	156,5	356,5	0,1

Здесь кроме t_0 и t_1 даны значения t_m – задержки при максимальной длине сегмента. Различают начальный, промежуточный и конечный сегмент. У них

разные значения t_0. Это объясняется тем, что отличаются функции хабов в этих сегментах. Все значения t_0, t_1 и t_m даны в *BT* (Bit Time – время передачи одного бита, для Ethernet 100 нс).

Суммарная величина задержки всех сегментов не должна превышать 575*BT* (PDV_{max} = 575*BT*) Кроме этого при расчете ЛВС вводят понятие запаса надежности

$SF=PDV_{max} - PDV$.

Минимальное значение *SF* чаще всего полагают равным

$SF_{min}=5\cdot BT$.

Рассмотрим 2 примера.

Пример 1. Рассчитать максимально возможную длину ЛВС на витой паре и двух хабах, исходя из условия обнаружения коллизий (рис. 3.1).

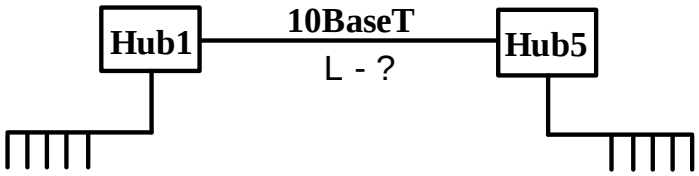


Рисунок 3.1.ЛВС на двух хабах

Здесь при расчете надо найти величину *L* – длину линии на витой паре, связывающей два хаба. Расстояние от хабов до сетевых станций берем максимальное

$$t_{m\ нач} + (t_{0\ пром}+t_1\cdot L) + t_{m\ кон} + SF_{min} = PDV_{max}. \qquad (3.1)$$

Подставляя значения параметров из таблицы 3.1 и беря значения PDV_{max}= = 575*BT* и SF_{min} = 5*BT*, найдем *L* = 2877 м.

Максимальная длина сети с учетом длины начального и конечного сегментов, равной по 100м, составит

L_{max} = 3077 м.

Полученное значение L_{max} не соответствует требованию стандарта 10BaseT, когда максимальный размер сети не должен превышать 500 м. Это ограничение связано с учетом затухания сигнала в линии. Чтобы удовлетворить это условие и увеличить длину *L* до расчетной величины, надо либо применить ВОЛС (стандарт 10BaseF), либо между хабами поставить повторители.

Пример 2. Рассчитать *PDV* для неоднородной ЛВС, содержащей 5 хабов (рис. 3.2).

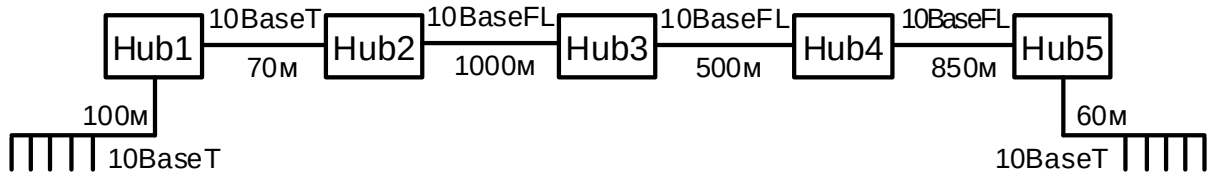


Рисунок 3.2. ЛВС на пяти хабах

Выполняя вычисления в соответствии с выражением (3.1) для длин кабеля каждого сегмента согласно рисунка 3.2 имеем

PDV = 26,6+42+0,113*70+33,5+0,1*1000+33,5+0,1*500+33,5+

+0,1*850+105+0.113*60+5 = 567 *BT*.

Полученная величина *PDV* = 567 *BT*. В примере правило четырех хабов не выполняется, но общая задержка меньше предельно допустимой. Такую ЛВС можно реализовать.

Кроме расчета *PDV* для ЛВС на хабах считают величину уменьшения межкадрового интервала *IPG*=96 *BT*. Этот интервал может сокращаться вследствие нестабильности задержки в хабах. Суммарное уменьшение задержки не должно превышать 47 *BT*. Обычно производители сетевого оборудования обеспечивают необходимую стабильность задержки, поэтому методику расчета здесь не приводим.

Расчет производительность сети Ethernet

При оценке производительности сети будем различать максимальную производительность (пропускную способность) и реальную производительность обмена между двумя станциями:

Максимальная производительность реализуется при работе в сети только двух абонентов с максимально возможной скоростью.

Производительность снижается за счет передачи:

– служебной информации (заголовков),

– межкадрового интервала (*IPG*).

Наибольшее влияние эти факторы оказывают на кадры минимальной длины (72 байта), так как удельный вес полезной информации (46 байт) при этом наименьший. В этом случае число n_k коротких кадров, передаваемых в секунду

$$n_k = \frac{1}{(72 \cdot 8 + 96) \cdot 10^{-7}} = 14880 \text{ кадров/с.}$$

Пропускная способность C_{nk} находится из учета только информационных байтов в пакете

$$C_{nk} = 14880 \cdot 46 \cdot 8 = 5{,}48 \text{ Мбит/с.}$$

Эффективность η_k передачи кадров минимальной длительности

$$\eta_k = \frac{C_{nk}}{C_{n\max}} = 0{,}548.$$

Для кадров максимальной длины (1500 байт) количество кадров n_g

$$n_g = \frac{1}{(1526 \cdot 8 + 96) \cdot 10^{-7}} = 812 \text{ кадров/с.}$$

Пропускная способность C_{ng} при передаче кадров максимальной длины

$$C_{ng} = 812 \cdot 1500 \cdot 8 = 9{,}76 \text{ Мбит/с,}$$

а эффективность η_g передачи кадров максимальной длины

$$\eta_k = \frac{C_{ng}}{C_{n\max}} = 0{,}976.$$

Таким образом, максимальная производительность для самых длинных кадров приближается к 100 %.

Реальная производительность. При малой интенсивности обмена одновременно работающих станций производительность обмена между станциями

делится пропорционально нагрузке. С ростом нагрузки (объема передаваемой информации) производительность вначале растет линейно, а затем начинают возникать коллизии и рост реальной производительности замедляется. При нагрузке сети более 0,7–0,8 возникает коллапс сети. Она не передает информацию, а занимается только устранением коллизий.

Fast Ethernet

Быстрое развитие технологии Ethernet, рост числа пользователей, развитие прикладных процессов, требующих передачи больших объемов информации (передачи файлов, видео и т.п.) с одной стороны и простота технологий, ее экономическая привлекательность с другой предопределили появление следующего более скоростного стандарта – Fast Ethernet со скоростью 100 Мбит/с (802.3u).

Само присутствие в названии новой технологии слова Ethernet означает высокую преемственность. Что сохранилось в Fast Ethernet:

1. Способ доступа CSMA/CD.
2. Форматы кадров.
3. Подуровни MAC и LLC.
4. Временные соотношения при обмене. Поскольку скорость передачи возросла в 10 раз, то в 10 раз уменьшилось значение биттайма *BT* = 10 нсек и всех остальных величин, связанных по времени. Наиболее серьезные ограничения касаются диаметра сети. Поскольку время двойного оборота сократилось до ~ 5,2 мкс, диаметр домена тоже уменьшился до 200 м.
5. Fast Ethernet также использует топологию «звезда» либо в полудуплексном режиме (с концентратором), либо в дуплексном режиме (с коммутатором).

Все изменения произошли на физическом уровне. Они вызваны тем, что при увеличении скорости передачи информации в 10 раз и сохранении длины сегмента 100 м, надо компенсировать увеличившиеся за счет расширения полосы частот затухание сигнала.

На физическом уровне различают 4 версии.

100BaseTX – наиболее популярная версия и реализуется на витой паре UTP-5, которая обеспечивает полосу пропускания 100 МГц и поэтому для Fast Ethernet достаточно две пары проводов. Разъемы соответствуют стандарту 10 Base T, что дает нужную совместимость Ethernet ~ Fast Ethernet. В качестве линейного кода использовался MLT-3, заимствованный из технологии FDDI. Код MLT-3 – трехуровневый, усложненный вариант NRZ ЧПИ. При передаче «0» значение не меняется, при передаче «1» значения меняются по цепочке +V, 0, –V, 0, +V (рис. 3.3).

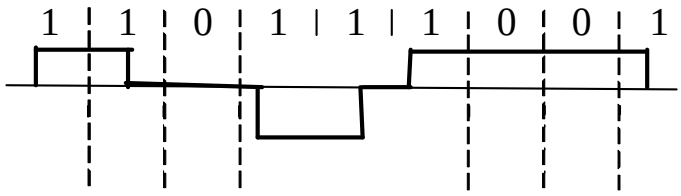


Рисунок 3.3. Код MLT-3

Этот код благодаря биполярному характеру сигнала и свойствам NRZ характеризуется уменьшенной шириной спектра.

Для логического кодирования применен код 4B/5B, который исключает наличие большого количества «0», следующих подряд. Суть этого кода поясняется в таблице 3.2.

Таблица 3.2. Пример кодирования 4B/5B

Входной сигнал	Выходной сигнал
0 0 0 0	1 1 1 1 0
0 0 0 1	0 1 0 0 1
0 0 1 0	1 0 1 0 0
. . .	. . .
1 1 1 1 1	1 1 1 0 1

Поскольку число комбинаций выходного кода гораздо больше, то «лишние» комбинации считают запрещенными и их появление сигнализирует об ошибках. Сочетание кодов MLT-3 и 4B/5B, по сравнению с манчестерским кодом обеспечивает при равной пропускной способности меньшую полосу пропускания в 8/5 раза. Это объясняется тем, что для манчестерского кода длительность минимального импульса равна половине тактового интервала, в то время как для MLT-3 они равны (выигрыш в 2 раза). Применение кода 4B/5B уменьшает требуемую скорость в 5/4 раза $(\frac{2}{5/4} = 8/5)$.

100BaseT4 – эта версия может быть реализована даже на витой паре категории 3 (UTP-3), несмотря на то, что частотный диапазон такого кабеля составляет всего 16 МГц. Для того, чтобы обеспечить скорость передачи 100 Мбит/с при такой полосе частот применяют следующие меры:

Однонаправленная параллельная передача сразу по 3 парам. При этом необходимая скорость передачи уменьшается до 33,3 Мбит/с.

Применение биполярного кода 8B/6T, когда комбинация двоичного кода из 8 бит передается 6 символами, имеющими 3 значения: «+V», «–V», и 0. Поскольку в выходном сигнале кодера длительность импульса возрастает в раза, то требуемая пропускная способность при физической скорости 33,3 Мбит/с достигает 25 Мбит. Другой важной особенностью сигнала 8B/6T является то, что его частотный спектр укладывается в полосу пропускания 16 МГц, что достаточно для кабеля UTP-3. Четвертая пара в кабеле UTP-3 используется для прослушивания сети. При появлении в ней сигнала при собственной передаче делается заключение о наличии коллизии.

Основным недостатком технологии 100BaseT4 является полудуплексным режим работы, а ее применимость объясняется тем, что она обеспечивает самый простой переход со стандарта 10BaseT.

100BaseFX – это версия для многомодового волокна с длиной волны мкм. Здесь применяется логическое кодирование 4B/5B, как и в 100 Base TX, и физический код NRZI. Передача и прием могут вестись как в полудуплексном, так и в полнодуплексном режимах по двум волокнам. В случае полудуплексного

режима размер сети ограничивается коллизиями – 412 м, а в режиме full duplex – затуханием (2 км для многомодового волокна и 32 км для одномодового). Переход со 100BaseFX на 10BaseFL невозможен, поскольку стандарт 10BaseFL рассчитан на $\lambda = 0{,}83$ мкм.

100BaseSX – версия на недорогих светодиодах ($\lambda = 0{,}83$ мкм) и многомодовом волокне. Это позволяет реализовать совместимость с 10BaseFL, но при этом уменьшается дальность до 300 м. Это дешевая альтернатива 100BaseFX.

Основные характеристики интерфейсов Fast Ethernet приведены в таблице 3.3.

Таблица 3.3. Характеристики стандарта Fast Ethernet

Характеристики	100BaseFX	100Base TX	100Base T4
Порт устройства	Duplex SC	RJ-45	RJ-45
Среда передачи	ВОЛС (мм)	UTP-5	UTP-3,4,5
Число пар (волокон)	2	2	4
Линейный код	NRZI	MLT-3	ЧПИ
Логический код	4B/5B	4B/5B	8B/6T
Максимальная длина сегмента	≤ 412 м (мм); ≥2 км (мм, FD)	100 м	100 м

Проектирование сетей Fast Ethernet в пределах домена коллизий (на хабах) также производится по двум моделям.

Модель 1. Эта модель не требует расчетов и предлагает проектировщикам 4 стандартных варианта.

Вариант 1А (без хаба). Здесь рассматривается соединение «точка-точка» двух узлов. В качестве узлов могут выступать рабочая станция, сервер, коммутатор. В этом случае необходимо обеспечить требования только по максимальному расстоянию (табл. 3.3, строка 6).

Вариант 1В. Этот вариант предполагает применение хаба класса 1 с задержкой на двойном пробеге не более 130 *BT*. В этих хабах допускаются порты T4, TX, FX. Поскольку из-за трансляции протоколов задержка достаточно большая, то может использоваться только один хаб, а расстояние от станций до хаба не более 100 м. При этом диаметр сети – до 200 м.

Вариант 1С. Здесь применяются хабы класса 2, у которых задержка по портам TX/FX – 46 *BT*, а для портов T4 – 33,5 *BT*. Поэтому допускается соединение двух хабов с расстоянием между ними 5 м. Максимальная длина сегментов на витой паре 100 м, на оптоволокне – 136 м.

Небольшое количество хабов не препятствует развитию сетей Fast Ethernet поскольку наряду с хабами используются коммутаторы, которые делят сеть на отдельные домены коллизий. Общая длина сети может быть в этом случае достаточно большой.

Модель 2. Эта модель, также как для Ethernet, предполагает расчет удвоенного времени задержки и сравнение его с предельным значением 512 *BT*. Эти задержки даны в таблице 3.4.

Таблица 3.4. Максимально допустимые задержки

Устройство / кабельный сегмент	Задержка на двойном пробеге, *BT*

Сетевая карта или порт коммутатора (два элемента): TX/FX T4 T4 с TX/FX	 100 138 127
Хаб класса 1	140
Хаб класса 2 (порты TX/FX)	92
Хаб класса 2 (порты T4)	67
Витая пара UTP-3, 1 м	1,14
Витая пара UTP-5, 1 м	1,112
Оптоволокно, 1 м	1,0

При расчете задержки рекомендуется оставить запас $SF = 4\ BT$.

Контрольные вопросы

Основные задачи, решаемые ЛВС

Типичное значение числа компьютеров, подключаемых к ЛВС

Какие топологии реализуется в ЛВС с хабом

Какие топологии реализуется в ЛВС с коммутатором

Раскрыть аббревиатуру CSMA/CD

Назначение MAC подуровня

Назначение LLC подуровня

Содержание адреса сетевой карты

Основные характеристики стандарта 10BaseT

Основные характеристики стандарта 10BaseF

Характеристики направляющих сред стандарта 10BaseXX

Факторы, ограничивающие диаметр сети Ethernet

Основные функции сетевой карты

Основные функции хаба

Достоинства и недостатки стандарта 10BaseFL

Типовые рекомендации ("правила") при построении однородных ЛВС

Сравнение пропускной способности сети 10BaseT для кадров максимальной и минимальной длины

Сравнить скорости передачи для кодов Манчестер и 4B/5B

Особенности временных параметров для Fast Ethernet

Основные модели построения сетей Fast Ethernet

Особенности расчета параметров сетей Fast Ethernet

3.2 Сетевой уровень. Адресация в IP сетях. Таблицы маршрутизации.

При подготовке к данному занятию использовать соответствующие разделы лекций и материалов, указанных в перечне литературы. Особое внимание обратить на следующие вопросы: адресация в IP сетях; типы адресов стека TCP/IP: локальные адреса, сетевые адреса, доменные имена; использование масок в IP адресации; взаимное отображение имен и адресов.

Компоненты заголовка IP пакета

Задание

Заголовок IP пакета представлен шестнадцатеричным кодом 0x4500002F116A00001E0612F4C02AFC01C02AFC14.

По принятой информации определить основные параметры служебной информации пакета, например, IP адрес узла назначения и протокол верхнего уровня, использующий данный пакет.

Решение

Во-первых, в принятой последовательности символов следует выделить компоненты кода, размещенные в полях заголовка IP пакета

«Протокол» и «IP адрес назначения».

Каждый байт заголовка представлен двумя символами шестнадцатеричного кода. Полю заголовка «Протокол» соответствует байт с номером 10, полю «IP адрес назначения» – байты заголовка с номерами от 17 до 20. После определения местоположение этих байтов получаем, что в этих полях размещены следующие сегменты кода заголовка:

0x06 и 0xC02AFC14.

Код адреса удобнее разбить на байты

0xC0, 0x2A, 0xFC, 0x14.

Это позволяет побайтно перевести шестнадцатеричный код в десятичный и записать IP адрес в десятичной нотации

192.42.252.20.

Сеть с таким адресом относится к сетям класса А.

И, наконец, шестнадцатеричный код 0x06 соответствует десятичному коду 06. Из таблицы 3.5 определяем, что этому коду в поле «Протокол» соответствует протокол транспортного уровня TCP.

В таблице 3.5 для сведения приведены номера некоторых протоколов, помещаемых в поле «Протокол» заголовка IP пакета. Полный перечень номеров протоколов опубликован в стандарте RFC 1700.

Коды остальных полей заголовка пакета определяются аналогичным образом.

Таблица 3.5. Коды протоколов

Код протокола	Название протокола	Назначение протокола
1	ICMP	Протокол контрольных сообщений [RFC 792]
2	IGMP	Групповой протокол управления [RFC 1112]
3	GGP	Протокол маршрутизатор-маршрутизатор [RFC-823]
4	IP	IP поверх IP (инкапсуляция/туннели)
5	ST	Поток [RFC 1190]
6	TCP	Протокол управления передачей [RFC-793]
8	EGP	Протокол внешней маршрутизации [RFC-888]
9	IGP	Протокол внутренней маршрутизации
10	BBN-MON	BBN-RCC мониторирование
11	NVP-II	Сетевой протокол для голосовой связи [RFC-741]

15	Xnet	Перекрестный сетевой отладчик [IEN158]
17	UDP	Протокол дейтограмм пользователя [RFC-768]
18	MUX	Мультиплексирование [IEN90]
19	DCN-MEAS	DCN измерительные субсистемы
20	HMP	Протокол мониторирования ЭВМ (host [RFC-869])
27	RDP	Протокол для надежной передачи данных [RFC-908]
28	IRTP	Надежный TP для Интернет [RFC-938]
29	ISO-TP4	ISO транспортный класс 4 [RFC-905]

Компоненты таблицы маршрутизации

Задание 1

Схема сети некоторой организации представлена на рисунке 3.4 и выглядит следующим образом:

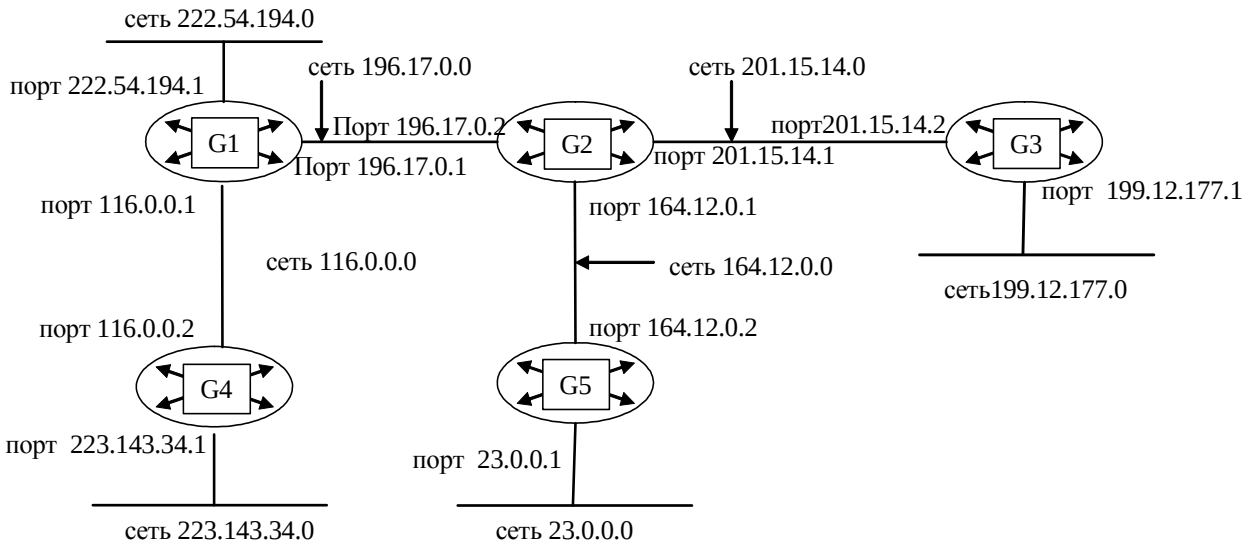


Рис.3.4. Схема корпоративной сети

Составьте таблицу маршрутизации для маршрутизатора G2, в которой укажите:

адреса всех сетей, входящих в составную сеть;

сетевой адрес следующего маршрутизатора, на который необходимо переслать пакет;

сетевой адрес выходного порта маршрутизатора G2;

расстояние до сети назначения (критерий выбора маршрута – количество пройденных в маршруте промежуточных маршрутизаторов).

Решение

Таблицу маршрутизации удобнее начинать с номеров сетей, непосредственно подключенных к данному маршрутизатору. Первые строчки таблицы в этом случае будут иметь вид (см. табл. 3.6):

Таблица 3.6. Первый этап заполнения таблицы маршрутизации

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения

196.17.0.0	---	196.17.0.2	0 (подсоединена)
201.15.14.0	---	201.15.14.1	0 (подсоединена)
...			
...			
...			

Затем следует поочередно описывать сети по мере удаления от данного маршрутизатора. В конце концов, для наиболее удаленных от маршрутизатора сетей будут сформированы записи вида (см. табл.3.7.):

Таблица 3.7. Завершающий этап заполнения таблицы маршрутизации

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
...			
...			
...			
203.143.34.0	196.17.0.1	196.17.0.2	2
199.12.177.0	201.15.14.2	201.15.14.1	1

Задание 2

Для структуризации составной сети используется 3 маршрутизатора. Составьте схему этой сети, если таблица маршрутизации одного из маршрутизаторов содержит следующие записи (табл.3.8):

Таблица 3.8. Исходная таблица маршрутизации

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
195.13.1323.0	198.37.6.1	198.37.6.2	1
197.18.11.0	----	197.18.11.1	0 (подсоединена)
198.37.6.0	----	198.37.6.2	0 (подсоединена)
205.36.11.0	198.37.6.1	198.37.6.2	1
213.14.46.0	----	213.14.46.1	0 (подсоединена)

Решение

Построение схемы сети также следует начинать с сетей, непосредственно подсоединенных к данному маршрутизатору. Затем поочередно подключают сети по мере их удаления от исходного узла. Окончательный вариант схемы рассматриваемой сети может иметь вид, приведенный на рисунке 3.5:

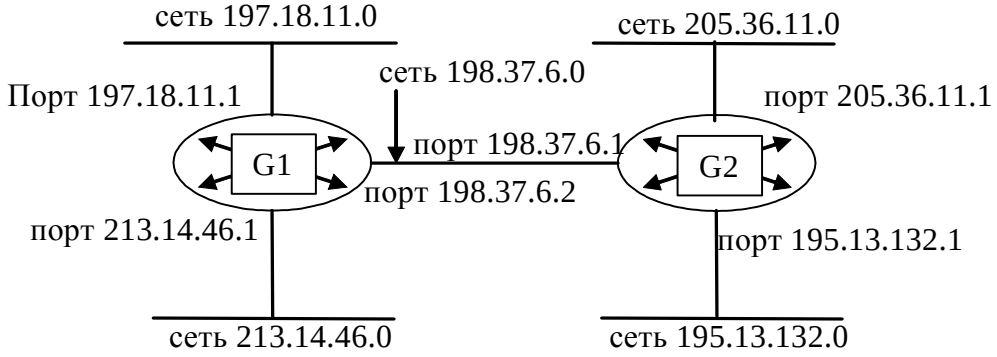


Рис.3.5. Возможный вариант структуры сети

Варианты комплексных заданий

Вариант1

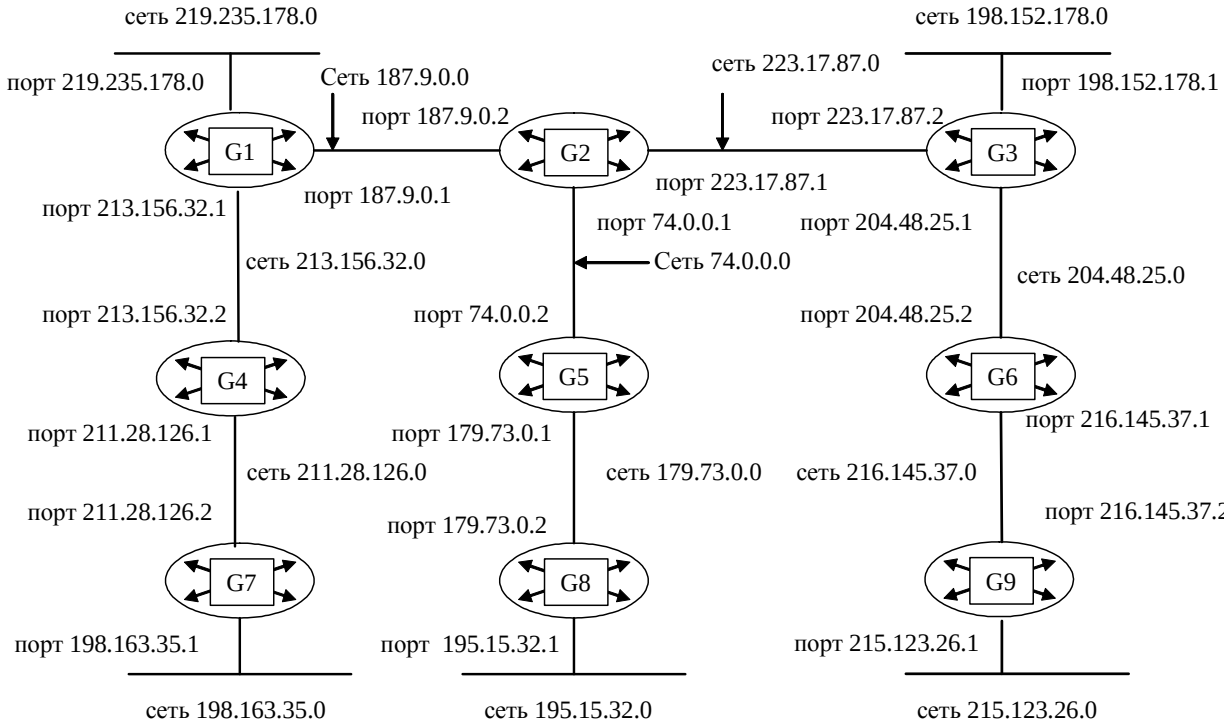
Задание 1

Заголовок IP пакета представлен шестнадцатеричным кодом 0x45000103116A000043111256C24A7C32C32B5D13.

По принятой информации определить параметр «Время жизни пакета» и IP адрес узла источника (в десятичной нотации).

Задание 2

Схема сети некоторой организации выглядит следующим образом:



Составьте таблицу маршрутизации для маршрутизатора G1, в которой укажите:

адреса всех сетей, входящих в составную сеть;

сетевой адрес следующего маршрутизатора, на который необходимо переслать пакет;

сетевой адрес выходного порта маршрутизатора G1;

расстояние до сети назначения (критерий выбора маршрута – количество пройденных в маршруте промежуточных маршрутизаторов).

Задание 3

Для структуризации составной сети используется 3 маршрутизатора. Составьте схему этой сети, если таблица маршрутизации одного из маршрутизаторов содержит следующие записи:

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
139.6.0.0	----	139.6.0.2	0 (подсоединена)
191.132.144.0	198.152.0.1	198.152.0.2	1
196.9.98.0	198.152.0.1	198.152.0.2	1
198.152.0.0	----	198.152.0.2	0 (подсоединена)
209.175.136.0	198.152.0.1	198.152.0.2	2
214.198.126.0	----	214.198.126.2	0 (подсоединена)

Вариант2

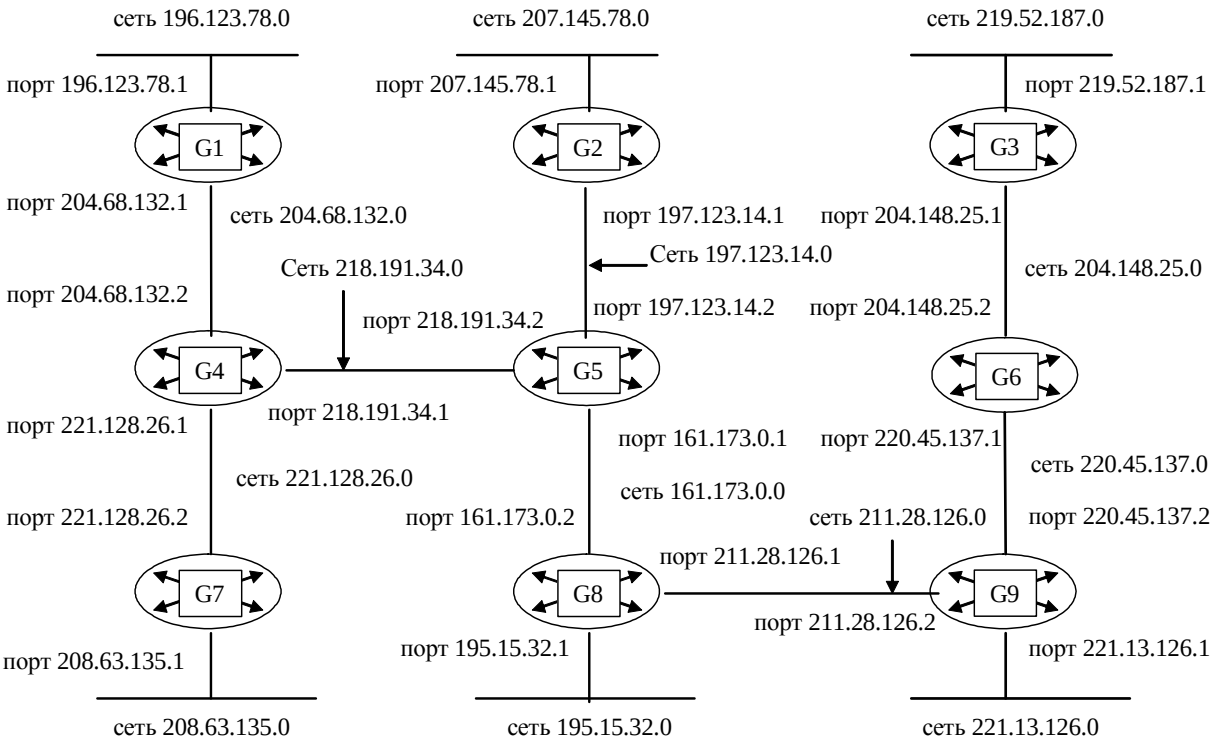
Задание 1

Заголовок IP пакета представлен шестнадцатеричным кодом 0x4500002320B3000067046007C1372B51C25D8B7A.

По принятой информации определить протокол верхнего уровня, использующий данный пакет, а также указать класс адресов сети, в которой расположен узел источника, и класс адресов сети, в которой расположен узел приемника.

Задание 2

Схема сети некоторой организации выглядит следующим образом:



Составьте таблицу маршрутизации для маршрутизатора G6, в которой укажите:

адреса всех сетей, входящих в составную сеть;

сетевой адрес следующего маршрутизатора, на который необходимо переслать пакет;

сетевой адрес выходного порта маршрутизатора G6;

расстояние до сети назначения (критерий выбора маршрута – количество пройденных в маршруте промежуточных маршрутизаторов).

Задание 3

Для структуризации составной сети используется 3 маршрутизатора. Составьте схему этой сети, если таблица маршрутизации одного из маршрутизаторов содержит следующие записи:

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
165.204.0.0	----	165.204.0.2	0 (подсоединена)
193.48.97.0	219.42.153.1	219.42.153.2	1
194.76.187.0	----	194.76.187.2	0 (подсоединена)
200.137.94.0	219.42.153.1	219.42.153.2	2
212.134.65.0	219.42.153.1	219.42.153.2	1
219.42.153.0	----	219.42.153.2	0 (подсоединена)

Вариант3

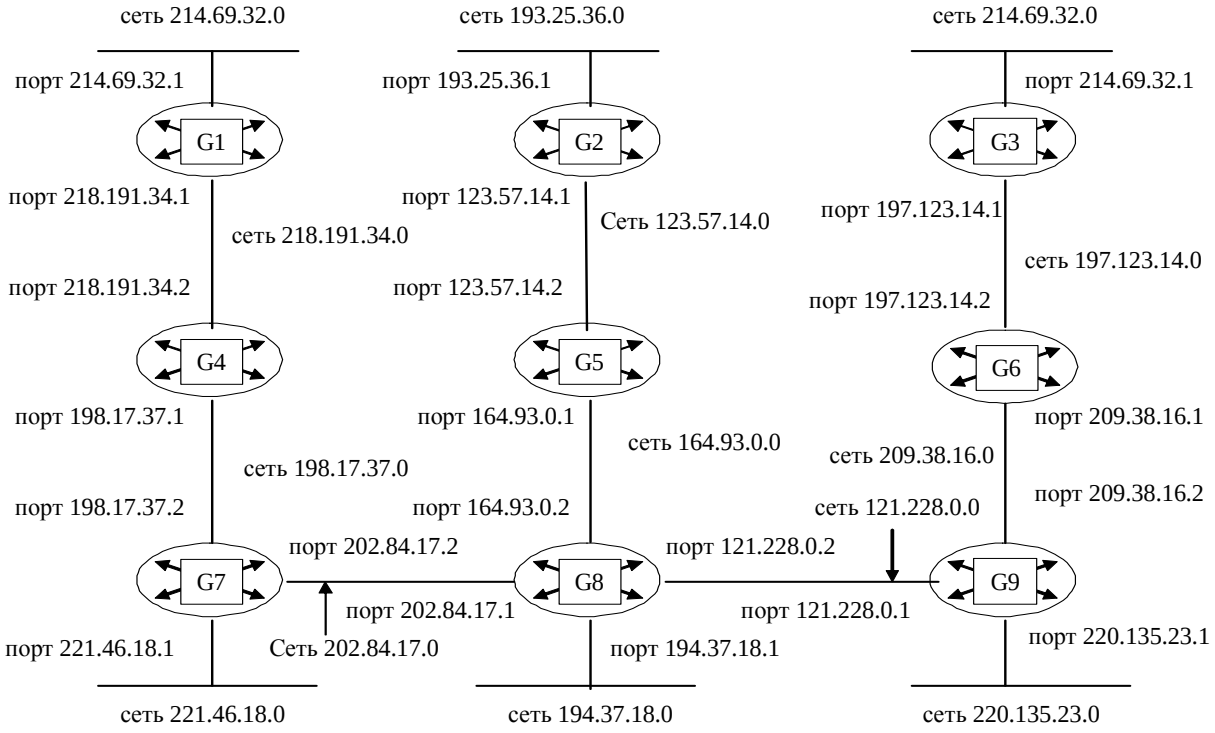
Задание 1

Заголовок IP пакета представлен шестнадцатеричным кодом 0x450001058B7A000089062B51AC137825A146C25D.

По принятой информации определить общую длину дейтаграммы и IP адрес узла назначения (в десятичной нотации).

Задание 2

Схема сети некоторой организации выглядит следующим образом:



Составьте таблицу маршрутизации для маршрутизатора G8, в которой укажите:

адреса всех сетей, входящих в составную сеть;

сетевой адрес следующего маршрутизатора, на который необходимо переслать пакет;

сетевой адрес выходного порта маршрутизатора G8;

расстояние до сети назначения (критерий выбора маршрута – количество пройденных в маршруте промежуточных маршрутизаторов).

Задание 3

Для структуризации составной сети используется 3 маршрутизатора. Составьте схему этой сети, если таблица маршрутизации одного из маршрутизаторов содержит следующие записи:

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
157.186.0.0	----	157.186.0.2	0 (подсоединена)
193.178.93.0	217.154.67.1	217.154.67.2	1
195.78.141.0	----	195.78.141.2	0 (подсоединена)
202.154.73.0	217.154.67.1	217.154.67.2	1
215.74.146.0	195.78.141.1	195.78.141.2	1
217.154.67.0	----	217.154.67.2	0 (подсоединена)

Контрольные вопросы и задания

Назначение сетевого уровня

Типы адресов в IP сетях

Локальные адреса в составных сетях

Адреса межсетевого уровня

Система доменных имен

Какую часть IP адресов занимают адреса класса A (B, C)?

Маски в IP сетях

Для заданной маски определить число возможных подсетей

Заголовок пакета сетевого уровня

Основы маршрутизации в IP сетях

Для приведенной схемы сети выбрать в таблице маршрутизатора M_n строку для сети назначения S_m

Фрагментация в IP сетях

3.3 Маршрутизация в IP-сетях. Использование масок

При подготовке к данному занятию использовать соответствующие разделы лекций и материалов, указанных в перечне литературы. Особое внимание обратить на следующие вопросы: принципы маршрутизации в IP сетях; протоколы маршрутизации в IP сетях.

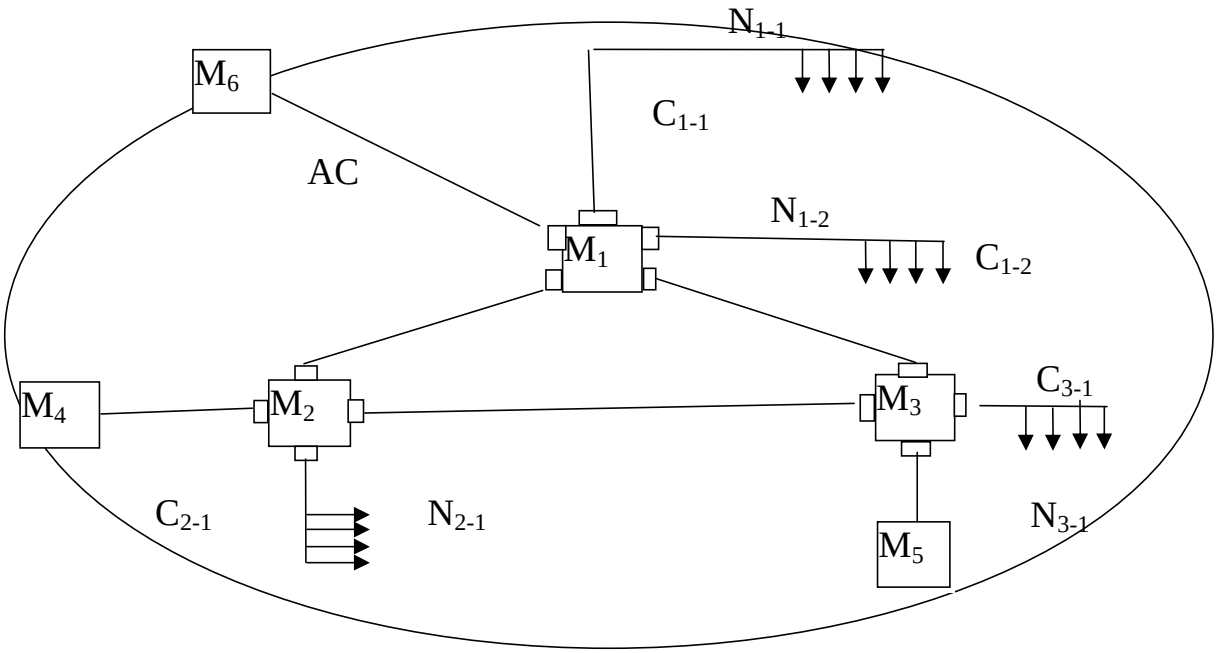


Рисунок 3.6. Структура автономной системы

Задание

Структура автономной системы (АС), представляющей сегмент IP сети, состоящей из маршрутизаторов M_i и подсетей C_{ik}, изображена на рисунке 3.6.

Адрес сети 128.12.0.0. Выход в глобальную сеть осуществляется через маршрутизатор M_n. Количество станций в подсетях N_{ik}.

Распределить адреса по подсетям, составить таблицу маршрутизации по протоколу RIPv2 для заданного маршрутизатора M_i. Варианты заданий приведены в таблице 3.9.

Таблица 3.9. Варианты параметров автономной системы

N варианта					i	n
1	2	3	4	5	1	4
6	7	8	9	10	2	5
100	150	200	300	500	N_{11}	
350	400	500	620	700	N_{12}	
150	370	50	250	120	N_{21}	
920	840	750	450	300	N_{31}	

Пример задания:

Таблица должна быть составлена для маршрутизатора M_1. Обеспечить выход во внешнюю сеть через маршрутизатор M_6. Параметры подсетей: N_{11}= 250, N_{12} = 180, N_{21} = 400, N_{31} = 800.

Решение.

1. Составим маски подсетей. В подсети C_{11}-число станций 250:

$2^6 = 128 < 250 < 256 = 2^8$.

Таким образом, маска будет содержать единицы в трех старших октетах:

11111111 11111111. 11111111. 00000000

или в десятичной нотации:

255. 255. 255. 0

В этом случае адрес подсети будет иметь вид

128.12.0.0

или с префиксом, указывающим размер поля номера сети

128.12.0./24

Аналогично находим для C_{12}:

Маска 255.255.255.128.

Адрес подсети составим так. Поскольку адресное поле 128.12.0.0 в четвертом байте уже почти полностью занято, возьмем адресное поле 128.12.1.0.

В этом поле будет задействовано только половина адресов от 128.12.1.0 до 128.12.1.127. Такие же операции выполним для подсетей N_{21} и N_{31} и получим следующие параметры. Для сети N_{21} маска имеет вид 255.255.254.0, диапазон адресов от 128.12.2.0 до 128.12.3.255. Для сети N_{31} маска имеет вид 255.255.252.0, диапазон адресов от 128.12.4.0 до 128.12.7.255.

2. Затем составим маршрутную таблицу для маршрутизатора M_1 по протоколу RIPv2.

Для этого снова модернизируем схему сети с адресами подсетей и портов маршрутизаторов. Адреса портов выбираем из адресного пространства той сети, которая подключена к этому порту. Так, порту 2 маршрутизатора (М1) назначим адрес 128.12.0.1. Соответственно, порту 3 маршрутизатора (М1) назначим адрес 128.12.1.6.

Для портов, соединяющих маршрутизаторы 4, 5, 6 нужно организовать свои подсети.

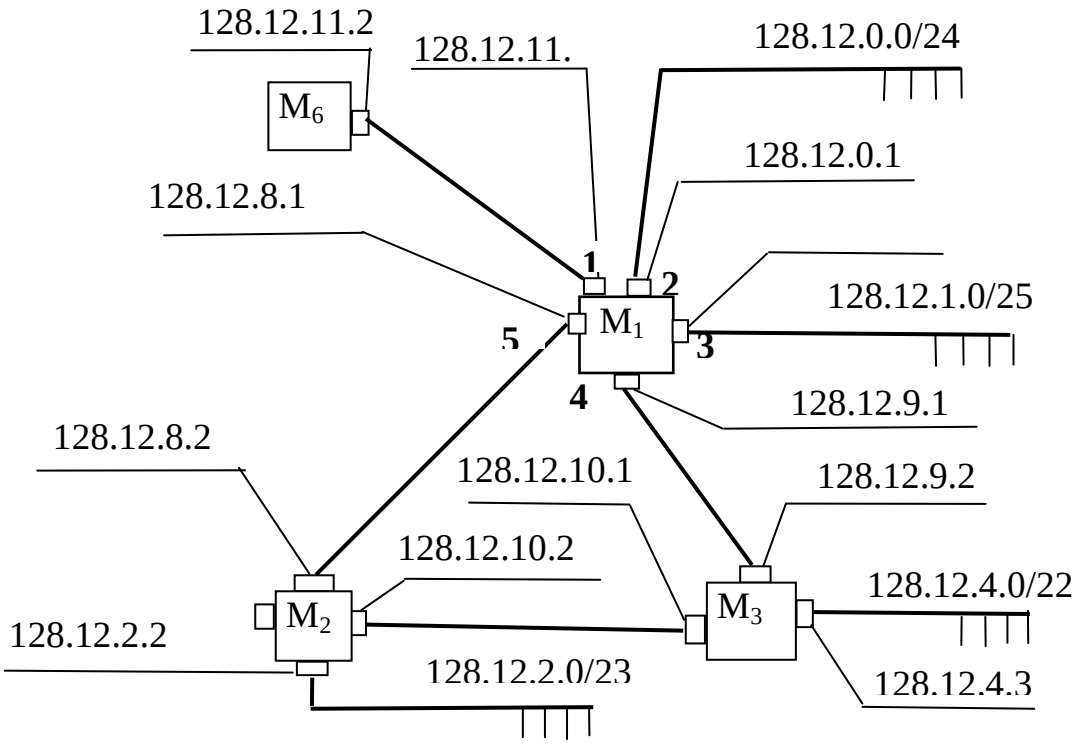


Рисунок 3.7. Расширенная структура автономной системы

Так для линии, соединяющей маршрутизаторы M_1 и M_2, минимально возможное адресное поле равно 4. Выберем подсеть 128.12.8.0 с префиксом 30 и

адреса портов 128.12.8.1 и 128.12.8.2. Аналогично для системы M_1–M_3 формируется подсеть 128.12.9.0/30 с адресами портов 128.12.9.1 и 128.12.9.2.

Для системы М2 –М3 создается подсеть 128.12.10.0/30 с адресами портов 128.12.10.1 и 128.12.10.2.

Система M_1–M_6 образует подсеть 128.12.11.0/30, адреса портов маршрутизаторов 128.12.11.1 и 128.12.11.2.

Перейдем к составлению таблицы.

Этап 1. Минимальная таблица учитывает только непосредственно подключенные подсети.

№ сети	Адрес следующего маршрутизатора	Порт	Метрика
128.12.0.0	128.12.0.1	2	0
128.12.1.0	128.12.1.6	3	0
128.12.9.0	128.12.9.1	4	0
128.12.8.0	128.12.8.1	5	0
128.12.11.0	128.12.11.1	1	0

Этап 2. Рассылка минимальных таблиц соседним маршрутизаторам.

Этап 3. Прием сообщений от соседей и пополнение таблицы. Маршрутизатор M_1 получает информацию от маршрутизаторов M_2 и M_3. и вносит ее в таблицу с коррекцией метрик до соответствующего маршрутизатора.

№	№ сети	Адрес следующего маршрутизатора	Порт	Метрика	Источник
1	128.12.0.0	128.12.0.1	2	0	
2	128.12.1.0	128.12.1.6	3	0	
3	128.12.9.0	128.12.9.1	4	0	
4	128.12.8.0	128.12.8.1	5	0	
5	128.12.11.0	128.12.11.1	1	0	
6	128.12.4.0	128.12.9.2	4	1	от M_3
7	128.12.9.0	128.12.9.2	4	1	от M_3
8	128. 12. 10.0	128.12.9.2	4	1	от M_3
9	128.12.2.0	128.12.8.2	5	1	от M_2
10	128. 12. 8.0	128.12.8.2	5	1	от M_2
11	128.12.10.0	128.12. 8.2	5	1	от M_2

В этой таблице есть повторяющиеся записи номеров сетей: строки с номерами 3 и 7 для сети 128.12.9.0 и строки с номерами 4 и 10 для сети 128.12.8.0. В этой таблице оставляем строки с номерами 3 и 4, так как у них лучшие метрики.

Записи строк с номерами 8 и 11 тоже одинаковы и имеют одинаковые метрики. Оставляем ту запись, которая появилась раньше, то есть строку с номером 8.

Для данной автономной системы формирование таблицы маршрутизации закончено. Для более крупной сети (автономной системы) процесс составления таблицы будет продолжаться до тех пор, пока в ней не будут перечислены все сети. На практике это означает отсутствие обновлений маршрутной информации при приеме очередной серии пакетов протокола маршрутизации.

Тестовые задания

Вариант 1

I. Планируется, что предприятие будет расширяться, и количество ПК в дальнейшем будет расти. Для упрощения администрирования расширяющейся сети класса А предполагается поделить ее на 6 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите возможное количество непомеченных бит в маске и узлов в каждой подсети.

II. Выберите маску, использующуюся по умолчанию для сетей класса С в двоичном и десятичном представлении.

III. Имеется: IP адрес: 222.27.147.198; маска подсети: 255.255.255.248. Укажите номер конечного узла.

IV. Вычислительную сеть предприятия необходимо подключить к глобальной сети. Для этого у провайдера был получен уникальный IP-адрес: 123.103.68.39. К какому классу относится данная сеть?

V. Предприятие имеет сеть класса С. Для упрощения администрирования расширяющейся сети предполагается поделить ее на 12 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите количество помеченных бит в маске (помеченными считаются биты с единичным значением) и саму маску в двоичном и десятичном представлении.

VI. Имеется: IP адрес: 201.26.63.206; маска подсети: 255.255.255.192. Укажите IP-адрес сети.

VII. Имеется: IP адрес: 203.204.47.93; маска подсети: 255.255.255.254. Укажите номер подсети.

VIII. У вас есть сеть класса В и 21-битовая маска подсети. Сколько подсетей и хостов вы получите?

IX. Ваша сеть класса А содержит 30 подсетей. В следующие два года вам необходимо организовать еще 50 подсетей, причем так, чтобы к каждой из них можно было подключить максимальное число хостов. Какую маску подсети следует выбрать?

X. Сеть 203.21.15.0 требуется разделить на 9 подсетей. При этом необходимо подключить к каждому сегменту максимально возможное число хостов. Какую маску подсети следует выбрать?

Вариант 2

I. Планируется, что предприятие будет расширяться, и количество ПК в дальнейшем будет расти. Для упрощения администрирования расширяющейся сети класса А предполагается поделить ее на 3 подсети. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите возможное количество непомеченных бит в маске и узлов в каждой подсети.

II. Выберите маску, использующуюся по умолчанию для сетей класса В в двоичном и десятичном представлении.

III. Имеется: IP адрес: 203.204.47.93; маска подсети: 255.255.255.254. Укажите номер конечного узла.

IV. Вычислительную сеть предприятия необходимо подключить к глобальной сети. Для этого у провайдера был получен уникальный IP-адрес: 222.73.177.56. К какому классу относится данная сеть?

V. Предприятие имеет сеть класса С. Для упрощения администрирования расширяющейся сети предполагается поделить ее на 20 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите количество помеченных бит в маске (помеченными считаются биты с единичным значением) и саму маску в двоичном и десятичном представлении.

VI. Имеется: IP адрес: 222.27.147.198; маска подсети: 255.255.255.248. Укажите IP-адрес сети.

VII. Имеется: IP адрес: 201.26.63.206; маска подсети: 255.255.255.192. Укажите номер подсети.

VIII. У вас есть сеть класса А и 19-битовая маска подсети. Сколько подсетей и хостов вы получите?

IX. Ваша сеть класса А содержит 60 подсетей. В следующие два года вам необходимо организовать еще 40 подсетей, причем так, чтобы к каждой из них можно было подключить максимальное число хостов. Какую маску подсети следует выбрать?

X. Сеть 192.168.1.0 требуется разделить на 9 подсетей. При этом необходимо подключить к каждому сегменту максимально возможное число хостов. Какую маску подсети следует выбрать?

Вариант 3

I. Планируется, что предприятие будет расширяться, и количество ПК в дальнейшем будет расти. Для упрощения администрирования расширяющейся сети класса А предполагается поделить ее на 19 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите возможное количество непомеченных бит в маске и узлов в каждой подсети.

II. Выберите маску, использующуюся по умолчанию для сетей класса В в двоичном и десятичном представлении.

III. Имеется: IP адрес: 201.26.63.206; маска подсети: 255.255.255.192. Укажите номер конечного узла.

IV. Вычислительную сеть предприятия необходимо подключить к глобальной сети. Для этого у провайдера был получен уникальный IP-адрес: 204.16.74.131. К какому классу относится данная сеть?

V. Предприятие имеет сеть класса В. Для упрощения администрирования расширяющейся сети предполагается поделить ее на 13 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите количество помеченных бит в маске (помеченными считаются биты с единичным значением) и саму маску в двоичном и десятичном представлении.

VI. Имеется: IP адрес: 203.204.47.93; маска подсети: 255.255.255.254. Укажите IP-адрес сети.

VII. Имеется: IP адрес: 222.27.147.198; маска подсети: 255.255.255.248. Укажите номер подсети.

VIII. У вас есть сеть класса A и 17-битовая маска подсети. Сколько подсетей и хостов вы получите?

IX. Ваша сеть класса B содержит 19 подсетей. В следующие два года вам необходимо организовать еще 15 подсетей, причем так, чтобы к каждой из них можно было подключить максимальное число хостов. Какую маску подсети следует выбрать?

X. Сеть 198.134.104.0 требуется разделить на 9 подсетей. При этом необходимо подключить к каждому сегменту максимально возможное число хостов. Какую маску подсети следует выбрать?

Вариант 4

I. Планируется, что предприятие будет расширяться, и количество ПК в дальнейшем будет расти. Для упрощения администрирования расширяющейся сети класса A предполагается поделить ее на 24 подсети. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите возможное количество непомеченных бит в маске и узлов в каждой подсети.

II. Выберите маску, использующуюся по умолчанию для сетей класса A в двоичном и десятичном представлении.

III. Имеется: IP адрес: 217.126.163.206; маска подсети: 255.255.255.240. Укажите номер конечного узла.

IV. Вычислительную сеть предприятия необходимо подключить к глобальной сети. Для этого у провайдера был получен уникальный IP-адрес: 213.216.174.16. К какому классу относится данная сеть?

V. Предприятие имеет сеть класса C. Для упрощения администрирования расширяющейся сети предполагается поделить ее на 7 подсетей. Определите маску, которую необходимо использовать для получения требуемого количества подсетей. Определите количество помеченных бит в маске (помеченными считаются биты с единичным значением) и саму маску в двоичном и десятичном представлении.

VI. Имеется: IP адрес: 197.123.47.93; маска подсети: 255.255.255.240. Укажите IP-адрес сети.

VII. Имеется: IP адрес: 205.127. 87.198; маска подсети: 255.255.255.224. Укажите номер подсети.

VIII. У вас есть сеть класса A и 17-битовая маска подсети. Сколько подсетей и хостов вы получите?

IX. Ваша сеть класса B содержит 19 подсетей. В следующие два года вам необходимо организовать еще 15 подсетей, причем так, чтобы к каждой из них можно было подключить максимальное число хостов. Какую маску подсети следует выбрать?

X. Сеть 207.32.21.0 требуется разделить на 9 подсетей. При этом необходимо подключить к каждому сегменту максимально возможное число хостов. Какую маску подсети следует выбрать?

Контрольные вопросы и задания

Основы маршрутизации в IP сетях

Протоколы внутренней маршрутизации

Основные принципы функционирования протоколов маршрутизации RIP

Этапы построения таблицы маршрутизации

Особенности протокола OSPF

Для заданной маски определить число возможных подсетей

Для приведенной схемы сети выбрать в таблице маршрутизатора M_n строку для сети назначения S_m

Какую часть IP адресов занимают адреса класса A, B, C?

3.4 Элементы диагностики сети

Целью данной работы является знакомство с командами операционной системы Linux, обеспечивающими диагностику сетевых компонентов, а также получение доступной рядовому пользователю (не обладающему правами root – администратора) информации о параметрах сети с помощью этих утилит.

Совместное функционирование сетевых компонентов обеспечивает сложный аппаратно-программный комплекс, как каждого отдельного узла, так и сети в целом. Для поддержания соответствующего качества работы сети используется мощный набор средств, отслеживающих как изменения топологии сети, так и изменения режимов работы узлов и отдельных программных модулей, связанные с неравномерным характером нагрузки. Кроме того, прикладным процессам доступен набор средств, обеспечивающих получение информации о сетевых компонентах.

Краткие сведения о сетевых утилитах диагностики

1. Утилита **users**

Команда **users** выводит пользовательские имена клиентов, зарегистрированных на данном узле к настоящему времени.

Синтаксис команды:

users [OPTION]... [*FILE*]

Опции:

--help

- отобразить эту справку и выйти

--version

- показать информацию о версии и выйти

Выводит список имен подключенных пользователей в соответствии с файлом *FILE*. Если *FILE* не указан, используется /var/run/utmp. Обычно как *FILE* используют /var/log/wtmp.

2. Утилита **who**

Команда **who** показывает пользователей, зарегистрированных на данный момент.

Синтаксис команды:

who [OPTION]... [*FILE* | ARG1 ARG2]

Опции:

- a, --all
- эквивалентно опциям -b -d --login -p -r -t -T -u.
- b, --boot
- время последней загрузки системы.
- d, --dead
- печатать мертвые процессы.
- H, --heading
- печатать строку с заголовками столбцов.
- l, --login
- печатать процессы входа в систему.
--lookup
- пытаться канонизировать имена узлов через DNS.
-m
- только имя узла и пользователя, подсоединенного стандартным вводом.
-p, --process
- печатать активные процессы, порожденные init.
-q, --count
- все имена и число подключенных пользователей.
-r, --runlevel
- печатать текущий уровень выполнения
-s, --short
- печатать только имя, линию и время. Установлено по умолчанию.
-t, --time
- печатать последнее изменение системного времени
-T, -w, --mesg
- добавлять статус приема сообщений как +, - или ?
-u, --users
- перечислить подключенных пользователей.
--message, -writable
- эквивалентно -T.
--help
- отобразить эту справку и выйти.
--version
- показать информацию о версии и выйти.

Возможен также вариант команды

who am i
- эквивалентно команде who -m.

Если FILE не указан, используется /var/run/utmp. Если заданы аргументы ARG1, ARG2, предполагается использование ‘am i’ или ‘mom likes’.

3 Утилита **hostname**

Команда **hostname** устанавливает или показывает системное имя локального компьютера (хоста). Устанавливать системное имя имеет право только администратор.

Синтаксис команды:

hostname [-v] [-a] [--alias] [-d] [--domain] [-f] [--fqdn] [-i] [--ip-address] [--long] [-s] [--short] [-y] [--yp] [--nis]

hostname [-v] [-F *filename*] [--file *filename*] [*hostname*]

hostname [-v] [-h] [--help] [-V] [--version]

Опции:

-a, --alias

- отобразить псевдоним узла (если используется).

-d, --domain

- отобразить имя домена системы DNS.

-f, --fqdn, --long

- отобразить полное доменное имя FQDN (Fully Qualified Domain Name) системы DNS. FQDN состоит из короткого имени хоста и имени домена DNS.

-F, --file filename

- чтение имени узла из указанного файла filename.

-h, --help

- вывести сообщение об использовании справочной системы и выйти из нее.

-i, --ip-address

- отобразить IP-адрес узла.

-s, --short

- отобразить короткое имя узла (имя хоста до первой точки).

-V, --version

- вывести информацию о версии программы и выйти из нее.

-v, --verbose

- установить подробный формат вывода сообщения.

-y, --yp, --nis

- отобразить имя домена NIS.

Когда команда вызывается, без аргументов программа отображает текущее имя узла.

4. Утилита **dnsdomainname**

Команда **dnsdomainname** выводит доменное имя DNS.

Синтаксис команды:

dnsdomainname [-v] [-h] [--help] [-V] [--version]

Опции команды dnsdomainname соответствуют опциям утилиты hostname. Фактически dnsdomainname дублирует команду hostname –d.

5. Утилита **host**

Host – утилита для выполнения поисков в системе DNS. Обычно используется для преобразования системного имени в IP адрес.

Синтаксис команды:

host [-aCdlriTwv] [-c *class*] [-N *ndots*] [-R *number*] [-t *type*] [-W *wait*] [-m *flag*] [-4] [-6] {*name*} [*server*]

Опции:

-a

- эквивалентно -v или -t опции.

-C

- сравнить записи SOA (Start of Authority) для имен зоны на авторитетных серверах имен.

-d

- эквивалентно -v опции (различия были в прежних версиях).

-l

- указать вид списка (символьные имена и числовые адреса). В комбинации с –a выводить все записи - список всех узлов домена, используя AXFR.

-r

- отмена режима рекурсии при запросе серверов имен, т. е. узлу разрешено имитировать поведение сервера имен без направления запросов другим серверам имен.

-i

- обратный поиск IP адреса по протоколу v6 должен использовать домен IP6.INT. По умолчанию использовать IP6.ARPA.

-T

- использовать TCP соединение для запроса. TCP автоматически выбирается для запросов, которые требуют это, таких как запросы передачи зоны (AXFR). По умолчанию использовать UDP.

-w

- всегда ожидать ответа на запрос.

-v

- разрешить подробный вывод.

-c *class*

- выполнить запрос класса *class* DNS. По умолчанию установлен класс IN (Internet).

-N *ndots*

- установить количество разделительных точек *ndots* в составном имени, для того, чтобы считать его абсолютным. По умолчанию это число определено спецификацией /etc/resolv.conf, либо равно 1, если ее нет. Имена с меньшим количеством точек интерпретируются как относительные и для их поиска используют дополнительные указатели в /etc/resolv.conf.

-t *type*

- выбрать тип запроса (CNAME, NS, SOA, SIG, KEY, AXFR, и т.д.). Хост автоматически выбирает требуемый тип запроса. По умолчанию представляются записи адреса.

-W *wait*

- установить время ожидания ответа на запрос *wait* секунд.

-R *number*

- установить количество попыток UDP запросов *number* для поиска. По умолчанию равно 1.

-m *flag*

- установка памяти, используемой для отладочных флагов (record | usage | trace).

name

- под именем *name* понимается доменное имя, которое должно быть найдено. Это также может быть IP адрес в децимальной точечной нотации версии IPv4 либо с разделительными двоеточиями версии IPv6.

server

- дополнительный аргумент в виде имени или IP адреса сервера имен, который узел должен запросить вместо серверов, указанных в спецификации etc/resolv.conf.

Когда команда вызывается без аргументов и опций, программа отображает краткое описание строки аргументов и опций команды.

6. Утилита **netstat**

Программа **netstat** распечатывает информацию о сетевых соединениях, таблицах маршрутизации, статистиках интерфейсов, замаскированных соединениях и групповых вещаниях.

Синтаксис команды:

netstat [*address_family_options*] [--tcp|-t] [--udp|-u] [--raw|-w] [--listening|-l] [--all|-a] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--symbolic|-N] [--extend|-e[--extend|-e]] [--timers|-o] [--program|-p] [--verbose|-v] [--continuous|-c] [*delay*]

netstat {--route|-r} [*address_family_options*] [--extend|-e[--extend|-e]] [--verbose| -v] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous| -c] [*delay*]

netstat {--interfaces|-i} [iface] [--all|-a] [--extend|-e[--extend|-e]] [--verbose|-v][--program|-p] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous|-c] [*delay*]

netstat {--groups|-g} [--numeric|-n] [--numeric-hosts][--numeric-ports][--numeric-ports] [--continuous|-c] [*delay*]

netstat {--masquerade|-M} [--extend|-e] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous|-c] [*delay*]

netstat {--statistics|-s} [--tcp|-t] [--udp|-u] [--raw|-w] [*delay*]

netstat {--version|-V}

netstat {--help|-h}

Описание команд:

Netstat распечатывает информацию о функционировании сетевых подсистем Linux. Тип выводимой информации определяет первый аргумент команды.

Без аргументов по умолчанию отображается перечень открытых сокетов.

Если не указано никакое адресное семейство address_family_options, выводится список активных сокетов всех конфигураций.

Первый параметр команды может принимать следующие значения

address_family_options:

[--protocol={inet,unix,ipx,ax25,netrom,ddp}[,...]]

либо [--unix|-x] [--inet|--ip] [--ax25] [--ipx] [--netrom] [--ddp]

- указать адресные семейства, поддерживающие соединения.

--route, -r

- отобразить ядро таблицы маршрутизации.

--interface=*iface* , -i

- печатать таблицу всех сетевых интерфейсов или указанных параметром *iface*.

--groups, -g

- отобразить информацию о членах мульти-вещательных групп для IPv4 и IPv6.

--masquerade, -M

- отобразить список замаскированных соединений.

--statistics, -s

- отобразить итоговую статистику для каждого протокола.

--version, -V

- показать информацию о версии и выйти.

--help, -h

- отобразить эту справку и выйти.

Дополнительные аргументы:

--tcp, -t

- отобразить активные соединения через сокеты tcp.

--udp, -u

- отобразить активные соединения через сокеты udp.

--raw, -w

- отобразить активные соединения через низкоуровневые сокеты raw.

--listening, -l

- отображать только прослушивающие сокеты (опущено по умолчанию).

--all,-a

- показать как прослушивающие, так и не слушающие сокеты. С опцией --interfaces показать интерфейсы, которые не отмечены.

--numeric, -n

- показать числовые адреса вместо попыток определения символьных имен узла, порта и пользовательского имени.

--numeric-hosts

- показать числовые адреса узлов, но не затрагивать разрешения порта или пользовательского имени.

--numeric-ports

- показать числовые номера портов, но не затрагивать разрешения имен узлов или пользовательских имен.

--numeric-users

- показать числовые идентификаторы пользователей, но не затрагивать разрешения имен узлов или пользовательских имен.

--symbolic, -N

- показать аппаратные имена.

--extend, -e

- отобразить дополнительную информацию. Для максимальной детализации используют эту опцию дважды.

--timers, -o

- включить информацию, относящуюся к сетевым таймерам.

--program, -p

- показать идентификационный номер и название программы, к которой принадлежит каждый сокет.

--verbose,-v

- подробно сообщать о подключениях, действующих на данный момент. Особенно печатать некоторую полезную информацию о несконфигурированных адресных семействах.

--continuous, -c

- печатать длительно выбранную информацию каждую секунду.

delay

- повторять цикл печати через каждые *delay* секунд.

--protocol=family, -A

- указать адресное семейство (возможно, лучше описанное как протокол нижнего уровня) для каждого подключения, которое будет показано. Под семейством понимается перечень ключевых слов семейства адресов, таких как inet, unix, ipx, ax25, netrom и ddp. Имеет тот же самый эффект, как использование опций --net, --unix (-), --ipx, --ax25, --netrom и --ddp.

Семейство адресов inet включает в себя сокеты tcp, udp и raw протоколов.

--fib, -F

- отображать маршрутную информацию из пересылаемой информационной базы (Forwarding Information Base) (по умолчанию)

--cache, -C

- отображать маршрутную информацию из кэша

--notrim, -T

- остановить подгонку длинных адресов.

--context, -Z

- если SELinux доступен, печатать в контексте SELinux.

7. Утилита **ping** (**ping6**)

Утилита **ping** (**ping6**) посылает ICMP запросы сетевым узлам. **Ping** использует дейтаграммы ECHO_REQUEST протокола ICMP для вызова пакетов ECHO_RESPONSE протокола ICMP от хостов и шлюзов. Дейтаграммы ECHO_REQUEST (‘‘pings’’) IP и ICMP содержат заголовок, сопровождаемый структурой временных интервалов, и затем произвольным числом дополняющих байт, используемых для заполнения пакета.

Синтаксис команды:

ping [-LRUbdfnqrvVaAB] [-c count] [-i interval] [-l preload] [-p pattern] [-s packetsize] [-t ttl] [-w deadline] [-F flowlabel] [-I interface] [-M hint] [-Q tos] [-S sndbuf] [-T timestamp option] [-W timeout] [hop ...] destination

Опции:

-a

- звуковой ping.

-A

- адаптивный ping. Межпакетный интервал адаптируется к времени кругооборота пакетов. Межпакетный интервал адаптируется к времени кругооборота пакетов, так что не более одной (или более, если установлена опция -preload) без

ответной попытки присутствует настоящий момент в сети. Для пользователей, не имеющих привилегий, минимальный интервал составляет 200 мс. Для сети с низкоуровневым временем оборота этот режим по существу соответствует потоковому режиму.

-b

- позволять "пингование" широковещательных адресов.

-B

- не позволять ping изменять адреса источника попыток. Адреса ограничены одним, выбранным при старте ping.

-c *count*

- остановить после посылки *count* количества пакетов ECHO_REQUEST. С опцией deadline ping ожидает *count* пакетов ECHO_REPLY до тех пор, пока не закончится время ожидания.

-d

- установить опцию SO_DEBUG для используемого сокета. По существу эта опция сокета не используется ядром LINUX.

-F *flow label*

- разместить и установить 20 битовую метку потока в пакете эхо-запроса (только ping6). Если значение равно нулю, ядро размещает метку потока произвольно.

-f

- потоковый ping. Для каждого посланного эхо-запроса печатается точка ".", тогда как печатается каждый принятый отклик эхо-ответа. Это обеспечивает быстрое отображение потерянных пакетов. Если интервал не задан, он устанавливается в нуль, и пакеты выводятся непосредственно по возвращению, либо 100 раз за секунду (наибольшее из этих условий).

-i *interval*

- ожидает *interval* секунд между отправлениями каждого пакета. По умолчанию нормальный интервал ожидания между пакетами 1 секунда, или без ожидания в потоковом режиме. Устанавливать интервал менее 0,2 секунды может только администратор.

-I *interface address*

- установить определенный интерфейс адреса источника. Аргументом может быть числовой IP адрес либо имя устройства. Эта опция необходима для "пингования" IPv6 локальных адресов.

-l *preload*

- если определен, ping посылает столько пакетов, не ожидая ответа. Только администратор может выбрать *preload* более 3.

-L

- запретить кольцевание (loopback) широковещательных пакетов. Этот флаг применим для "пингования" широковещательных адресов назначения.

-n

- вывод только числовой. Не пытаться заставлять искать символические имена для адресов узлов.

-p *pattern*

- можно определить до 16 дополняющих байт для заполнения посылаемого пакета. Это полезно для диагностики проблем сети, зависящих от данных. Например, опция -p *ff* заставляет посылать пакеты, полностью заполненные единицами.

-Q *tos*

- установить биты дейтаграммы ICMP, относящиеся к назначению качества сервиса. Параметр *tos* может быть как десятичным, так и шестнадцатеричным числом. Традиционно (RFC1349) интерпретируется как: 0 - зарезервирован (в настоящее время переопределен для контроля перегруженности), 1-4 для установления типа сервиса и 5-7 для установления приоритета. Доступные установки типа сервиса: минимальная стоимость - 0x02, надежность - 0x04, пропускная способность - 0x08, низкая задержка - 0x10. Много бит качества не должно быть установлено одновременно. Возможные параметры, определенные для диапазона приоритета - от приоритета (0x02) до сетевого контроля (0xe0). Нужно быть администратором (CAP_NET_ADMIN возможности) для использования критических или более высоких значений приоритета. Нельзя установить бит 0x01 (зарезервированный), если в ядре недоступен ECN. В REF2474 эти поля переопределены как 8-битовый дифференцированный сервис (DS), состоящий из: бит 0-1 - для разделения данных (нужно использование ECN), и бит 2-7 - основные коды Codepoint (DSCP).

-q

- ограниченный вывод. Ничего не отображать, кроме строк старта и завершения.

-R

- записать маршрут. Включить опцию RECORD_ROUTE в пакет эхо-запроса и отображать маршрутный буфер возвращаемого пакета. Заметим, что размеры IP заголовка достаточны лишь для 9 пунктов маршрута. Многие узлы игнорируют или сбрасывают эту опцию.

-r

- миновать обычную таблицу маршрутизации и отправлять непосредственно узлу по подключенному интерфейсу. Если узел не находится в непосредственно подключенной сети, возвращается ошибка. Эта опция может быть использована для "пингования" местных узлов по интерфейсу, через который нет маршрута, при условии, что предусмотрено также использование опции -I.

-s *packetsize*

- отправляет количество байт данных для передачи. По умолчанию - это 56 байт, которые транслируются в 64 байта данных IP, объединенных с 8 байтами заголовка ICMP.

-S *sndbuf*

- установить размер буфера сокета. Если не определено, выбирается буферизировать не более одного пакета.

-t *ttl*

- установить IP время жизни *ttl*.

-T *timestamp option*

- установить специальную IP опцию временная метка *timestamp*. Опция *timestamp* может выглядеть как *tsonly* (только временная метка), *tsandaddr* (временная метка и адрес) или *tsprespec host1* [*host2* [*host3* [*host4*]]] (временная метка и предусмотренные узлы)

-M *hint*

- выбрать стратегию выбора пути для максимальной размера блока передачи данных (MNU). Аргумент *hint* может принимать значение *do* (запрещает фрагментацию, даже локальную), *want* (выполнять стратегию поиска, фрагментировать локально, когда размер пакета больше), или *dont* (не устанавливать флаг DF).

-U

- печатать полное время ожидания пользователь-пользователь. Обычно ping печатает сетевое время кругооборота.

-v

- выводить подробную информацию.

-V

- показать версию и выйти

-w *deadline*

- определить задержку в секундах до завершения функционирования ping независимо от количества принятых или отправленных пакетов. В этом случае ping не останавливается после получения *count* пакетов, а ожидает также истечения последнего срока или ответа счетчика попыток или извещения об ошибке из сети.

-W *timeout*

- время *timeout* ожидания ответа в секундах. Опция влияет только при отсутствии любых ответов, иначе ping ожидает двойное время кругооборота.

Дополнительные сведения по использованию программы **ping**

При использовании утилиты **ping** для выделения ошибок он сначала должен быть запущен на местные узлы, чтобы проверить, что локальный сетевой интерфейс поднят и выполняется. Затем "пингуемые" хосты и маршрутизаторы должны быть продвинуты все дальше и дальше. Вычисляется время кругооборота и статистика потерь. Если получены копии пакетов, они не включаются в расчет потерь пакетов, хотя время кругооборота используется в вычислениях минимального, среднего и максимального значений времени кругооборота. Когда определенное количество пакетов будет отправлено (и получено) или если программа завершается с SIGINT, отображаются краткие итоги. Короче, текущая статистика может быть получена без завершения работы процесса с сигналом SIGQUIT.

Если **ping** не получает никаких ответных пакетов, он будет завершен с кодом 1. Если определены число пакетов *count* и последний срок *deadline*, а число принятых пакетов на интервале крайнего срока меньше, чем *count*, также будет выход с кодом 1. По другой ошибке - выход с кодом 2. В противном случае - выход с кодом 0. Это делает возможным использовать код выхода для того, чтобы видеть, работоспособен хост или нет.

Эта программа предназначена для использования при тестировании, измерении и управлении в сети. Поскольку ее загрузка может накладываться на сеть, неразумно использовать **ping** в течение обычного функционирования или из автоматизированных сценариев.

Детали пакета **ICMP**

Заголовок IP пакета без опций - 20 байт. ICMP пакет эхо-запроса содержит дополнительные важныt 8 байт ICMP заголовка, сопровождаемых произвольным количеством байт данных. Когда задан размер пакета, он указывает размер этой дополнительной части данных (по умолчанию - это 56). Таким образом, количество принимаемых данных IP пакета типа ICMP ECHO_REPLY будет всегда на 8 байт больше, чем затребовано объемом данных ICMP заголовка.

Если объем данных менее размера структурного интервала. ping использует начальные байты этого пространства для включения метки времени, которая используется для вычислении времени кругооборота. Если объем данных короче, время кругооборота не задано.

Продублированные и поврежденные пакеты

ping сообщает о продублированных и поврежденных пакетах. Дублирования пакетов не должно происходить никогда и, видимо, вызвано несоответствующей ретрансляцией пакетов на сетевом уровне. Дублирование может случаться во многих ситуациях и редко (если когда-либо) - хороший знак, хотя наличие низкоуровневых копий не всегда может быть причиной тревоги.

Поврежденные пакеты очевидно, более серьезная причина для тревоги и часто указывают неисправность аппаратных средств где-то на пути пакета ping (в сети или на узлах).

Затруднения, вызванные различными форматами данных

Межсетевой уровень никогда не должен обрабатывать пакеты, по разному зависимые от информации, содержащейся в разделе данных. К сожалению известно, что проблемы зависимости от данных прокрадываются в сеть, продолжительный период времени оставаясь не обнаруженными. Во многих случаях особенности форматов, которые будут иметь проблемы, это такие, как недостаточное количество переключений, например, длинные серии нулей или единиц.

Это означает, что если есть проблемы, связанные с зависимостью от данных, вероятно, придется выполнить большой объем тестирования, чтобы найти их. В случае удачи можно устраивать поиск файла, который каждый не может быть послан через сеть, или который требует много дольше для передачи, чем другой файл подобной длины. Потом можно исследовать этот файл для повторения образца, который можно тестировать, используя опцию -p команды ping.

Детали поля TTL

Значение TTL IP пакета представляет максимальное число IP маршрутизаторов, через которые пакет может пройти до того, как будет отброшен. В настоящей практике можно ожидать, что каждый маршрутизатор в интернет уменьшает поле TTL точно на 1.

Спецификация TCP/IP указывает, что поле TTL для пакета TCP должно быть установлено на 60, но многие системы имеют меньшие значения (4.3BSD использует 30, 4.2BSD использует 15).

Максимально возможное значение этого поля - 255 и наибольшая установка поля TTL Unix систем пакетов ICMP ECHO_REQUEST - 255. Вот почему можно "пинговать" некоторые хосты, но не достигать их по протоколам telnet или ftp.

При нормальном функционировании ping печатает значение TTL из принимаемых пакетов. Когда удаленная система получает пакет ping, она может выполнять одно из трех действий с полем TTL в своем ответе:

● Не изменять его, как это делала система Berkeley Unix до выпуска 4.3BSD Tahoe release. В этом случае значение TTL в принятом пакете будет равно 255 минус количество маршрутизаторов в маршруте кругооборота.

● Установить его на 255, как это делают современные системы Berkeley Unix. В этом случае значение TTL в принятом пакете будет равно 255 минус количество маршрутизаторов в маршруте от удаленной системы до "пингующего" узла.

● Установить ему несколько другое значение. Некоторые машины используют то же самое значение для ICMP пакета, которое они используют для TCP пакета, например, 30 или 60. Другие могут использовать полностью неконтролируемые значения.

Проблемы

Многие хосты и маршрутизаторы игнорируют опцию RECORD_ROUTE.

Максимальная длина IP заголовка слишком мала, чтобы были полностью использованы опции типа RECORD_ROUTE.

Потоковое "пингование" не рекомендуется, в общем случае, а потоковое "пингование" широковещательных адресов должно выполняться только при тщательно контролируемых условиях.

8. Утилита **tracepath** (**tracepath6**)

Команда **tracepath** (**tracepath6**) трассирует путь к сетевому узлу, раскрывая MTU вдоль этого пути.

Синтаксис команды:

tracepath *destination* [*port*]

Утилита **tracepath** устанавливает маршрут к сетевому узлу *destination* с определением максимального размера пакета. Она использует UDP порт *port* или какой-нибудь произвольный порт. Программа работает подобно утилите traceroute, но не требует привилегий суперпользователя и не предполагает опций.

Информация, выводимая в результате выполнения утилит

Информация, отображаемая утилитой **host**

HEADER

ЗАГОЛОВОК (поля выходных данных).

opcode

- код операции (например, запрос - QUERY).

status
- состояние (например, без ошибок - NOERROR).
id
- идентификатор.
flags
- флаги (например: qr, aa, rd, ra; соответственно: QUERY, ANSWER, AUTHORITY, ADDITIONAL).
QUESTION (ANSWER, AUTHORITY) SECTION
СЕКЦИЯ ОПРОСА (ОТВЕТА, ПОЛНОМОЧИЙ).
IN
- класс адресов - IP.
Типы запросов:
A
- преобразование имени в адрес (address) узла.
AAAA
- полное имя узла.
ANY
- отображение всех ресурсных записей.
CNAME
- псевдоним узла (canon name).
HINFO
- информация об аппаратном обеспечении узла.
MX
- информация о почтовом сервере (mail exchanger).
NS
- информация о сервере DNS.
PTR
- указатель преобразования IP адреса в имя узла.
SOA
- начало полномочий (Start of Authority).

Информация, отображаемая утилитой **netstat**
OUTPUT
ВЫВОД (поля выходных данных):
Active Internet connections (TCP, UDP, raw)
Активные соединения интернет (tcp, udp, raw):
Proto
- протокол (tcp, udp, raw) используемый сокетом.
Recv-Q
- количество байт, не скопированных пользовательской программой, подключенной к этому сокету.
Send-Q
- количество байт, не подтвержденных удаленным узлом.
Local Address

- адрес и номер порта сокета местного окончания. Если не указана опция --numeric (-n) – адрес сокета определен как каноническое имя узла (FQDN), а номер порта переведен в имя соответствующей службы.

Foreign Address

- адрес и номер порта сокета удаленного окончания. Аналогично с Local Address.

State

- состояние сокета. Пока не установлен статус в необработанном режиме и обычно нет состояний, используемых UDP, эта колонка может оставаться незаполненной. Обычно это может принимать одно из нескольких значений:

ESTABLISHED

- сокет имеет установленное соединение.

SYN_SENT

- сокет активно пытается установить соединение.

SYN_RECV

- из сети получен запрос на соединение.

FIN_WAIT1

- сокет закрыт и соединение разрывается.

FIN_WAIT2

- соединение закрыто и сокет ожидает закрытия от удаленного узла.

TIME_WAIT

- сокет после закрытия находится в состоянии ожидания пакетов, все еще находящихся в сети.

CLOSED

- сокет не может быть использован.

CLOSE_WAIT

- удаленный узел закрыт, сокет ожидает закрытия.

LAST_ACK

- удаленный узел закрыт, сокет закрыт и ожидает подтверждения.

LISTEN

- сокет на прослушивании поступающих соединений. Такие сокеты не включают в вывод (OUTPUT), если не установлены опции --listening (-l) или --all (-a).

CLOSING

- оба сокета закрыты, но все еще не получены все отправленные пакеты.

UNKNOWN

- состояние сокета неизвестно.

User

- имя или идентификатор (User ID) пользователя сокета.

PID/Program name

- разделенная слэшем пара - идентификатор процесса (Process ID) и имя процесса, которому принадлежит пакет. Опция --program заставляет включить эту колонку. Также нужно иметь привилегии суперпользователя, чтобы видеть информацию на сокете, который не принадлежит вам. Эта идентификационная информация недоступна для IPX сокетов.

Timer

- пока не записано.

Active UNIX domain Sockets

- активные сокеты домена UNIX

Proto

- протокол (обычно unix)

RefCnt

- счетчик ссылок (т. е. подключенных процессов через этот сокет).

Flags

- отображены флаги SO_ACCEPTON (показаны как ACC), SO_WAITDATA (W) или SO_NOSPACE (N). SO_ACCECPTON использованы на неподключенных сокетах, если соответствующие им процессы ожидают запроса на соединение. Другие флаги обычно не представляют интереса.

Type

- есть несколько типов доступа сокетов:

SOCK_DGRAM

- сокет, используемый в дейтаграммном режиме (без установления соединения).

SOCK_STREAM

- потоковый сокет (с установлением соединения).

SOCK_RAW

- сокет используется как низкоуровневый сокет.

SOCK_RDM

- сокет обслуживания сообщений надежной доставки.

SOCK_SEQPACKET

- сокет последовательности пакетов.

SOCK_PACKET

- сокет доступа низкоуровневых интерфейсов.

UNKNOWN

- неизвестный режим.

State

- состояние (содержит одно из ключевых слов).

FREE

- сокет не размещен.

LISTENING

- сокет, прослушивающий запросы на соединение. Такие сокеты включают в вывод (OUTPUT), если определены опции -listening (-l) или --all (-a).

CONNECTING

- сокет в состоянии установления подключения.

CONNECTED

- сокет подключен.

DISCONNECTING

- сокет в состоянии процесса разъединения.

(empty)

- (пустой) сокет ни к чему не подключен.

UNKNOWN

- такое состояние не должно никогда случаться.

PID/Program name:

- идентификатор и имя процесса, который имеет открытый сокет. Больше информации о секции активных соединений интернет написано выше.

Path

- имя маршрута, который соответствует процессу, подключенному к сокету.

Iface

- имя интерфейса.

MTU

- максимальный размер пакета в байтах (Maximum Transfer Unit).

RX-OK (TX-OK)

- количество принятых (отправленных) пакетов без ошибок.

RX-ERR (TX-ERR)

- количество принятых (отправленных) пакетов с ошибками.

RX-DRP (TX-DRP)

- количество отброшенных пакетов при приеме (отправке).

RX-OVR (TX-OVR)

- количество пакетов, потерянных при из-за переполнения при приеме (отправке).

Flg

- флаги (могут принимать значения):

B

- разрешить широковещательную передачу (broadcast);

L

- интерфейс обратной петли (loopback);

R

- интерфейс запущен;

U

- интерфейс активен.

Информация, выводимая утилитой **tracepath**

Первая колонка выводимой информации показывает TTL пробного пакета, сопровождаемая двоеточием. Значение TTL обычно получено из отклика сети, но иногда ответ не содержит необходимой информации и заставляет предполагать. В этом случае число сопровождается знаком вопроса (?).

Вторая колонка показывает сетевые этапы, которые были ответом на пробные пакеты; это - любой адрес маршрутизатора или слово [LOCALHOST], если пробный пакет не был отправлен в сеть.

Строки показывают разнообразную информацию о пути и соответствующих сетевых этапах. Как правило, они содержат значение среднего времени кругооборота RTT. Дополнительно может быть показан путевой MTU, когда он изменен. Если путь асимметричен или пробный пакет финиширует до достижения заданного этапа, показано различие между числом этапов в прямом и обратном направлении, сопровождаемое ключевым словом async. Такая инфор-

мация ненадежна. Третья строка показывает асимметрию с первой, так как первый пробник с TTL 2 был отброшен на первом этапе раскрытия пути.

На последней строке просуммирована информация о всем пути к узлу назначения, показан определенный путевой MTU, число этапов до цели и наши предположения о числе этапов от пункта назначения до нас, которое может отличающимся при асимметричном пути.

Практическое задание

1. Ознакомиться с описанием сетевых утилит, используемых в работе, по рекомендуемым в списке литературы документам и данному руководству.

2. Войти в режим командной строки. С помощью утилиты *users* определить список пользователей данного узла.

Открыть еще одну вкладку *Konsole* и повторить команду *users*. Зафиксировать изменения.

3. С помощью утилиты *who* установить время загрузки системы, исследовать процессы входа в систему, процессы, выполняемые на данном узле, и уже законченные (отобразить заголовки столбцов).

4. С помощью утилиты *hostname* определить короткое имя узла, имя домена. полное имя узла, IP адрес, псевдоним узла.

5. С помощью утилиты *host* вывести и проанализировать полную информацию для соседнего узла с именем tor0002X, в котором символ X принимает значение, отличающееся на единицу от последней цифры имени местного узла.

Провести анализ выводимой информации в зависимости от устанавливаемых флагов в пакетах запроса.

6. С помощью утилиты *netstat* проверить состояние сети. По таблице маршрутизации определить параметры подключения сети: адреса (маски) шлюзов, используемые интерфейсы.

По таблице сетевых интерфейсов определить максимальные размеры и статистики принятых пакетов всех подключений.

По таблицам активных соединений через сокеты UDP и TCP определить адреса и состояния соответствующих подключений.

Ознакомиться таблицей активных подключений через сокеты Unix. Определить режим доступа к сокету.

Ознакомиться со списком сокетов сервера, установленных на прослушивание.

Провести анализ устанавливаемых флагов в заголовках пакетов и выводимой информацией.

Просмотреть таблицу сокетов с дополнительной информацией о программе, использующей сокет. Определить идентификаторы программы и состояние подключения.

Определить групповые адреса узла для протоколов IPv4 и IPv6.

Ознакомиться с полной таблицей всех подключений (опции -v, -e). Установить различие выводимой информации для этих опций.

7. С помощью утилиты *ping* определить доступность узлов сети. Командой *ping* направить 3 пакета (не забывать ограничивать количество отправляе-

мых пакетов) по петле обратной связи (по собственному адресу). Провести анализ полученной информации.

Повторить выполненный анализ для соседнего узла сети, для публичного сервера ТУСУРа а также для узлов, указанных преподавателем.

Изменить в пакете количество пересылаемых байт данных. Убедиться о выполнении команд по выводимой информации.

Определить маршрут следования пакета до сервера ТУСУРа. Установить значение поля "ttl" равным количеству узлов маршрута. Отправить пакеты и провести анализ полей принятого пакета.

Уменьшить значение поля "ttl" на единицу. Повторить запрос. Сравнить результаты последних этапов исследований.

Установить режим регистрации временных меток. Провести анализ полученных результатов.

8. С помощью утилиты *tracepath* определить маршруты до исследованных в п. 7 задания узлов сети. Сравнить результаты проведенных испытаний. Оценить пропускную способность исследованных маршрутов и их временные характеристики.

9. С помощью пунктов меню *Konsole* "Сеанс/Печать экрана" сохранить протокол работы в файле *user_name*_lab1.txt (печатать в файл), где - *user_name* - имя пользователя.

10. В отчете по работе предоставить результаты исследований по тестированию сети и краткие выводы по каждому пункту задания. При защите работы уметь выполнить тестирование сетевых параметров, предложенных преподавателем, и проанализировать информацию, выводимую в результате выполнения соответствующей утилиты.

Контрольные вопросы

1 Исправить синтаксис команды $ *users -help*.

2. Для чего нужна утилита *users*?

3. В чем разница выполнения команд $ *who -H* и $ *who -p*?

4. Указать неверный параметр команды $ *who -login*.

5. Для чего нужна утилита *who*?

6. В чем разница выполнения команд $ *hostname -d* и $ *dnsdomainname -v*?

7. Указать неверный параметр команды $ *hostname -login*.

8. Для чего нужна утилита *hostname*?

9. В чем разница выполнения команд $ *host -a* и $ *host -v*?

10. Указать неверный параметр команды $ *host -l*.

11. Для чего нужна утилита *host*?

12. Вывод каких типов информации можно установить утилитой *host*?

13. Для чего нужна петля обратной связи?

14. Как проверить работоспособность петли обратной связи?

15. В чем разница выполнения команд $ *netstat -V* и $ *netstat -v*?

16. Выбрать формат команды *netstat* для определения групповых адресов.

17. Перечислить состояния сокетов, выводимые программой *netstat*.

18. Вывод каких режимы доступа к сокету предусмотрен программой *netstat*?

19. Какую информацию о статистике принятых пакетов предоставляет *netstat*?

20. Указать неверный параметр команды $ *netstat -l*.

21. Для чего нужна утилита *netstat*?

22. Вывод каких типов информации можно установить утилитой *netstat*?

23. Какой результат выдаст утилита *netstat* с параметром *–r*?

24. Какой протокол использован для работы с утилитой *ping*?

25. В чем разница выполнения команды *ping* при использовании ключей -R и -r?

26. Какая информация выводится в результате выполнения программы *ping*?

27. Какой протокол использован для работы с утилитой *tracepath*?

28. В чем разница выполнения команды *ping -R* и команды *tracepath*?

29. Какая информация выводится в результате выполнения программы *tracepath*?

30. Указать неверный параметр команды $ *tracepath -h*.

3.5 Удаленный доступ. Безопасность работы в сети

Целью данной работы является знакомство с протоколом SSH, обеспечивающими безопасное зашифрованное соединение двух ненадежных узлов через небезопасную сеть для регистрации пользователя на удаленном узле, а также получение навыков работы пользователей, не имеющих привилегий администратора, в режиме удаленного доступа.

Краткие сведения о пакете OpenSSH

Удаленный доступ традиционно является одним из основных компонентов сетевого взаимодействия. Например. в защищенных сетях в свое время широкое применение получил протокол telnet. Telnet позволяет пользователю установить TCP соединение с сервером и работать с ним по командам клиентской машины так, словно она является удаленным терминалом сервера. Однако проблемы безопасности ограничивают его использование в сетях, ненадежных с точки зрения безопасности.

В настоящее время в таких сетях широкое применение получает протокол ssh. Ssh (secure shell - безопасная оболочка) является протоколом для удаленного безопасного входа и других сетевых сервисов безопасности в недостаточно надежно защищенной сети. В общей архитектуре протокола ssh можно выделить три основных блока: протокол транспортного уровня, протокол аутентификации пользователя и протокол соединения.

Протокол транспортного уровня (SSH-TRANS) обеспечивает аутентификацию сервера, конфиденциальность и целостность соединения. Также может дополнительно обеспечивать сжатие данных. Протокол транспортного уровня обычно выполняется поверх соединения TCP, но может использоваться и поверх любого другого надежного соединения.

Протокол аутентификации пользователя (SSH-USERAUTH) аутентифицирует клиента для сервера. Он выполняется поверх протокола транспортного уровня.

С целью повышения безопасности соединения осуществляется как аутентификация клиента для сервера, так и аутентификация сервера для клиента. Клиент посылает запрос на обслуживание всякий раз, когда устанавливается безопасное соединение на транспортном уровне. Второй запрос сервиса посылается после выполнения аутентификации пользователя.

Протокол соединения (SSH-CONN), мультиплексирует несколько логических каналов в один зашифрованный туннель. Протокол выполняется поверх протокола аутентификации пользователя.

Протокол соединения создает каналы, которые могут использоваться для различных целей. Существуют стандартные методы установки безопасных сессий интерактивного shell и перенаправления ("туннелирования") произвольных портов TCP/IP и соединений X11.

На любом узле сети, использующем ssh, может выполняться как клиентская, так и серверная часть программы. Для работы с ssh каждый хост может иметь не менее одного ключа хоста, причем в шифровании могут быть использованы различные криптографические алгоритмы. Хосты могут иметь несколько ключей, используемых различными алгоритмами. Несколько узлов могут использовать общий ключ хоста. Однако каждый сервер должен иметь, по крайней мере, один ключ для каждого обязательного алгоритма открытого ключа. В настоящее время требуется поддерживать алгоритм DSS (Digital Signature Standard).

Ключ хоста-сервера используется при обмене открытыми ключами для проверки истинности соединения с подлинным (а не подмененным) сервером. Для этого клиент должен предварительно знать об открытом ключе сервера. Это знание может быть реализовано в рамках одной из двух моделей.

В первой клиент имеет локальную базу данных (файл), в которой каждому имени сервера ставится в соответствие его открытый ключ. Этот метод не требует централизованной административной инфраструктуры и трехсторонней координации. В то же время, такую базу данных тяжело поддерживать при большом количестве клиентов и серверов, с которыми они должны взаимодействовать.

Во второй модели вводится понятие доверенного сертификационного агента, который и отвечает за проверку соответствия имени хоста его открытому ключу. При этом упрощается поддержка клиента (он должен знать открытый ключ только самого сертификационного агента), но предъявляются высокие требования к сертификационному агенту, который должен иметь открытые ключи всех хостов, к которым обращаются клиенты.

Предусмотрена также возможность отказа от проверки открытого ключа хоста сервера при первом соединении клиента с сервером. Это обеспечивает возможность взаимодействия без предварительного знания ключа сервера. Такое соединение также обеспечивает защиту от пассивного прослушивания; но оно уязвимо для активных атак типа встреча посередине (man-in-the-middle), то

есть попытки временной подмены сервера. И такие соединения не должны быть допущены по умолчанию, если в сети допускается возможность активных атак. Однако, так как инфраструктура открытого ключа еще недостаточно широко распространена, данная опция делает протокол более приемлемым для взаимодействия сторон, обеспечивая более высокий уровень безопасности, чем такие предыдущие решения как telnet или rlogin.

В результате использования ssh, то есть выполнения процедуры проверки полномочий пользователей и последующего шифрования любой проходящей через сеть информации, от паролей до сеансов, обеспечивается более высокая степень защищенности удаленного доступа.

Компоненты службы удаленного доступа ssh (OpenSSH) реализуются различными командами. Например, клиентская программа ssh обеспечивает регистрацию и выполнение команд на удаленной машине. программа sshd (демон ssh) обеспечивает защищенное шифрованное соединение между двумя ненадежными узлами по небезопасной сети, программа ssh-keygen генерирует и управляет ключами идентификации. Кроме перечисленных, имеется и ряд иных утилит: scp, sftp и др.

Перенаправление произвольного TCP соединения через защищенный канал может быть определено как в командной строке, так и в файле конфигурации. Одно возможное приложение перенаправления TCP - это защищенное соединение с почтовым сервером, другое действует через сетевой экран.

Например, рассмотрим кодированное соединение между клиентом и сервером IRC, не имеющих непосредственной поддержки шифрованного соединения. Это действует следующим образом. Пользователь подключается к удаленному узлу, используя ssh, определяя порт, который должен быть использован для перенаправления соединения на удаленный сервер. После этого имеется возможность запустить службу шифрования на клиентской машине, подключенной к этому самому порту и ssh будет кодировать и перенаправлять соединение.

В следующем примере прокладывается туннель сеанса IRC от клиентской машины “127.0.0.1” (localhost) до удаленного сервера “server.example.com”:

```
$ ssh -f -L 1234:localhost:6667 server.example.com sleep 10
$ irc -c ’#users’ -p 1234 pinky 127.0.0.1
```

Здесь используется туннельное соединения IRC сервера “server.example.com”, связанного каналом с пользователем “#users”, nickname “pinky” с применением порта 1234. Несущественно, какой номер порта использован, насколько больше, чем 1023 (напомним, что только root может открывать сокеты на привилегированных портах), и не создается конфликтов с любым уже используемым портом. Соединение перенаправлено на порт 6667 удаленного сервера, поскольку это стандартный порт для IRC сервиса.

Опция -f устанавливает фоновый режим ssh и дистанционная команда “sleep 10” определяет разрешенное количество времени (10 секунд в примере) до старта службы, которая выполняет туннелирование. Если в пределах заданного времени соединение не сделано, ssh завершается.

Краткие сведения по командам ssh

SSH client - программа удаленного входа в систему.

Синтаксис команды:

ssh [-1246AaCfgkMNnqsTtVvXxY] [-b *bind_address*] [-c *cipher_spec*] [-D [*bind_address*:]*port*] [-e *escape_char*] [-F *configfile*] [-i *identity_file*] [-L [*bind_address*:]*port*:*host*:*hostport*] [-l *login_name*] [-m *mac_spec*] [-O *ctl_cmd*] [-o *option*] [-p *port*] [-R [*bind_address*:]port:host:*hostport*] [-S *ctl_path*] [-w *local_tun*[:*remote_tun*]] [*user@*]*hostname* [*command*]

Описание программы:

SSH client это программа для регистрации на удаленном узле и для выполнения команд на удаленной машине. Это предполагает замену программ **rlogin** и **rsh** и обеспечивает безопасное зашифрованное соединение двух ненадежных узлов через небезопасную сеть. Соединения X11 и произвольные TCP порты также могут пересылать безопасно.

SSH подключает и регистрирует к определенному узлу (с опцией имени пользователя). Пользователь должен доказать свою подлинность удаленной машине, используя один из несколько методов в зависимости от используемой версии протокола.

Если определена команда, она выполняется на уделенном узле, вместо входа в систему.

Опции

-1

- заставляет ssh использовать протокол только версии 1.

-2

- заставляет ssh использовать протокол только версии 2.

-4

- заставляет ssh использовать только адресацию IPv4.

-6

- заставляет ssh использовать только адресацию IPv6.

-A

- разрешить предварительное соединение агента аутентификации (это также может быть определено в файле конфигурации хоста).

-a

- запретить предварительное соединение агента аутентификации.

-b *bind_address*

- использовать *bind_address* на локальной машине как исходный адрес соединения. Использовать только в системах с более, чем одним адресом.

-C

- запрашивать сжатие всех данных, включая stdin, stdout, stderr, и данные для пересылки соединений X11 и TCP. Алгоритм компрессии тот же, который использует gzip.

-c *cipher_spec*

- выбрать спецификацию шифра для кодирования сессии. Протокол версии 1 допускает единственный перечень шифра. Поддерживаемые значения: “3des”, “blowfish”, “des”. По умолчанию используется “3des”. Для протокола версии 2

перечень cipher_spec представляет собой разделенный запятыми список в порядке предпочтения: 3des-cbc, aes128-cbc, aes192-cbc, aes256-cbc, aes128-ctr, aes192-ctr, aes256-ctr, arcfour128, arcfour256, arcfour, blow-fish-cbc, and cast128-cbc. По умолчанию: aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128, arcfour256,arcfour,aes192-cbc,aes256-cbc,aes128-ctr, aes192-ctr,aes256-ctr.

-D [*bind_address*:]*port*

- определить местный динамически перенаправляемый порт уровня приложений (привилегия root).

-e *escape_char*

- установить escape-символ для сеансов с **pty** (интерфейс псевдо терминала) (по умолчанию "~"). Escape-символ распознается только в начале строки. Escape-символ, следующий за точкой ("."), закрывает соединение, следующий за control-Z - приостанавливает соединение, и следующий непосредственно за собой, пересылает один escape-символ. Установка символа "none" отменяет любые escape-символы и делает сеанс полностью прозрачным.

-F *configfile*

- определить альтернативный файл конфигурации пользователя. Если файл конфигурации задан в командной строке, будет проигнорирован общесистемный файл конфигурации (/etc/ssh/ssh_config). По умолчанию: ~/.ssh/config.

-f

- запросить ssh перейти в фоновый режим перед выполнением команды. Это полезно, если ssh выполняет запрос пароля или передаваемой фразы.

-g

- позволять удаленному узлу подключить местный перенаправляемый порт.

-I *smartcard_device*

- определить устройство ssh, которое должно быть использовано для связи со смарткартой, используемой для записи приватных RSA ключей пользователя (если доступна поддержка таких устройств).

-i *identity_file*

- выбрать файл, из которого считывать идентичность (приватные ключи) для RSA или DSA аутентификации. По умолчанию ~/.ssh/identity для протокола версии 1 и ~/.ssh/id_rsa и ~/.ssh/id_dsa для протокола версии 2. Эти файлы могут быть также определены в файле конфигурации.

-k

- запретить перенаправление (делегирование) удостоверения GSSAPI серверу.

-L [*bind_address*:]*port*:*host*:*hostport*

- определить, что данный порт *port* на локальном (клиентском) узле пересылает информацию данным хосту *host* и порту *hostport* на удаленной стороне. Это выполняет распределение сокетов на прослушивание портов на ближней стороне, дополнительно связанных с определенным *bind_address*. Всякий раз, когда выполняется подключение к этому порту, соединение перенаправляется по защищенному каналу и соединяет с портом *hostport* узла *host* удаленной машины. Порт перенаправления также может быть определен в файле конфигурации.

-l *login_name*

- определить имя пользователя для регистрации на удаленной машине. Это также может быть определено в файле конфигурации хоста.

-N

- не выполнять удаленные команды (полезно только для перенаправления по протоколу версии 2).

-n

- переназначить **stdin** из /dev/null (предотвратить чтение из **stdin**).

-O *ctl_cmd*

- управлять мультиплексированием активных соединений основного процесса. Когда опция -O определена аргумент *ctl_cmd* интерпретируется и проходит в основной процесс. Правильные команды: “check” (проверять, что выполняет основной процесс) и “exit” (запрос выхода).

-o *option*

- задать опции в формате, используемом в файле конфигурации. Это полезно для определения опций, у которых нет отдельного флага в командной строке.

-p *port*

- задать номер порта *port* для соединения на удаленный хост. Это может определено в файле конфигурации хоста.

-q

- установить режим тишины. Заставляет запретить все предупреждающие и диагностические сообщения.

-R [*bind_address*:]*port*:*host*:*hostport*

- определить, что данный порт на удаленном (серверном) хосте перенаправлен на данные хост и порт на ближней стороне. Это работает локализацией сокета на прослушивание на удаленной стороне, и всякое подключение к этому порту перенаправляется по секретному каналу и подключается к порту *port* узла *host* локальной машины.

Порт перенаправления также может быть задан в конфигурационном файле. Привилегированные порты могут быть перенаправлены только когда на удаленной машине зарегистрирован суперпользователь. Адресация IPv6 может быть определена заключением в квадратные скобки, или использованием альтернативного синтаксиса: [*bind_address*/]*host*/*port*/*hostport*.

По определению сокеты на прослушивание на сервере ограничены только интерфейсом обратной связи. Это может быть проигнорировано определением *bind_address*. Пустой *bind_address* или адрес ‘*’ указывают, что удаленный сокет может прослушивать все интерфейсы. Указание на удаленный *bind_address* будет успешным, если только доступна опция сервера *GatewayPorts*.

-S *ctl_path*

- определить позицию управляющего сокета для разделения соединения. Ссылки на описание *ControlPath* и *ControlMaster* смотреть на *ssh_config*(5).

-s

- разрешить использование запроса вызова подсистемы на удаленной системе. Подсистемы представлены протоколом **ssh2**, которые способствуют использованию **ssh** как безопасного транспорта для других приложений (например, sftp). Подсистема определяется как дистанционная команда.

-T

- запретить назначение псевдо терминала.

-t

- назначить псевдо терминал. Это может быть использовано для выполнения произвольных экранных программ на удаленной машине, которое может быть очень полезным, в том числе, когда обслуживается выполнение меню. Многократные опции -t усиливают назначение псевдо терминала, даже если ssh не имеет локального tty.

-V

- отобразить номер версии и выйти.

-v

- режим подробного вывода. Заставляет ssh выводить отладочные сообщения о ходе выполнения. Они полезны для отладки проблем соединения, аутентификации и конфигурации. Многократные опции -v увеличивают многословие. Максимум - 3.

-w *local_tun*[:*remote_tun*]

- запросить перенаправление туннельных устройств, определенных командой **tun** между клиентом (*local_tun*) и сервером (*remote_tun*).

-X

- Разрешить перенаправление X11. Это также может быть определено в файле конфигурации. Пересылка X11 должно быть доступна с осторожностью. Пользователи, полномочные миновать разрешающий файл на удаленном узле (для базы данных разрешенных X-пользователей) могут иметь доступ к локальному X11 дисплею через пересылаемое соединение. Атакующий может затем выполнить такие действия, как мониторинг нажатия клавиш. По этой причине пересылка X11 подвергается преобразованию X11 SECURITY.

-x

- запретить перенаправление X11.

Практическое задание

1. Ознакомиться с описанием протокола **ssh** и утилит, используемых в работе (упомянутых ниже в пунктах задания), по рекомендуемым в списке литературы документам, данному руководству и справочной системы Linux.

2. Войти в режим командной строки. С помощью команды **ssh** подключиться к узлу, с которым работали по п. 5 предыдущей работы. На соответствующий запрос сервера ввести пароль для регистрации на удаленной машине.

3. Ознакомиться с файловой системой сервера. С помощью команды **pwd** определите имя текущего каталога. С помощью команды **ls** определите содержимое текущего каталога.

С помощью команд **cd** и **ls** ознакомиться с содержанием соседних каталогов.

4. С помощью команды **mkdir** домашнем каталоге создать именной подкаталог. В этом подкаталоге с помощью команды **touch** создать текстовый файл с именем lab_1.

5. С помощью текстового редактора (например, **ed** или **vim**), войти в режим редактирования этого файла и внести в него следующую информацию:

Фамилию ИО (студента);

группу, факультет;

имя пользователя;

сетевое имя локального узла;

IP адрес и маску локального узла;

имя домена;

адрес шлюза по умолчанию.

6. С помощью утилиты **users** определить список пользователей удаленного узла.

7. С помощью утилиты **who** установить время загрузки системы, исследовать процессы входа в систему, процессы, выполняемые на данном узле, и уже законченные.

8. С помощью утилиты **hostname** определить короткое имя узла, имя домена. полное имя узла, IP адрес, псевдоним узла.

9. С помощью утилиты **host** вывести и проанализировать полную информацию для локального узла (на котором работаете непосредственно в данный момент).

10. С помощью утилиты **netstat** проверить состояние сети. По таблице маршрутизации определить параметры подключения сети: адреса (маски) шлюзов, используемые интерфейсы.

11. Наиболее внимательно проанализировать информацию по соединению **ssh** на ближнем и удаленном узле.

12. С помощью утилиты **ping** определить доступность узлов сети. Командой **ping** направить 3 пакета локальному узлу.

Повторить операцию для публичного сервера ТУСУРа.

13. С помощью утилиты **tracepath** определить маршрут до узла, заданного преподавателем. Оценить пропускную способность маршрута и временные характеристики.

14. С помощью пунктов меню **Konsole** "Сеанс/Печать экрана" сохранить протокол работы на удаленной системе в файле lab_2.txt (печатать в файл). В отчете также предоставить результаты редактирования текстового файла. При защите работы уметь выполнять команды навигации по удаленной системе, созданию и перемещению файлов и каталогов, редактирования текстовых файлов.

Контрольные вопросы

1 Для чего нужна программа ssh?

2. Что общего и в чем отличие ssh и telnet?

3. Основные достоинства и недостатки telnet.

4. Основные компоненты ssh.

5. Назначение протокола транспортного уровня ssh.

6. Основные характеристики протокола аутентификации пользователя ssh.

7. Назначение протокола соединения ssh.
8. Что понимается под ключом хоста?
9. Как проверяется подлинность сервера?
10. Что такое сертификационный агент?
11. Особенности режима работы ssh без проверки полномочий сервера.
12. Что понимается под TCP перенаправлением?
13. Как реализуется перенаправление TCP соединения?
14. Записать обязательные элементы команды ssh.
15. Привести формат команды ssh при заданных портах местного и удаленного узла.
16. Для чего нужна команда pwd?
17. Какие ключи используют с командой cd?
18. Что выполняется по команде mkdir?
19. Для чего используется команда ls?
20. Назначение утилиты ed.
21. Для чего нужна утилиты vim?

3.6 Исследование сетевых компонентов с помощью имитатора Net-simulator

При подготовке к данному занятию использовать соответствующие разделы лекций и материалов, указанных в перечне литературы. Особое внимание обратить на следующие вопросы: реализация межсетевого взаимодействия средствами TCP/IP; типы адресов стека TCP/IP; взаимное отображение имен и адресов; использование масок в IP адресации; средства построения составных сетей.

Целью занятия «Исследование сетевых компонентов с помощью имитатора Net-Simulator» является знакомство с сетевым имитатором Net-Simulator, предназначенным для имитационного моделирования процессов, происходящих в сетях связи, а также получения первичных навыков конфигурирования сетевых интерфейсов.

Краткие сведения о сетевом имитаторе Net-Simulator

Net-Simulator является программным продуктом, позволяющим осуществить имитационное моделирование сетей ЭВМ, имеет открытую лицензию и предназначен для учебных целей. Исходные материалы продукта размещены на сайте http://sourceforge.net/projects/net-simulator/. Копии основных документов этого проекта продублированы в каталоге S:\БогомоловСИ\Model\NetSim\Net-Simulator.

Net-Simulator позволяет строить виртуальные вычислительные сети из виртуальных сетевых устройств: маршрутизаторов, настольных компьютеров, концентраторов и т.п. Устройствами можно управлять при помощи интерфейса командной строки из виртуальных терминалов. В виртуальных сетях реализованы канальный и сетевой уровни в соответствии с ISO OSI, что достаточно для первичного обучения конфигурированию и поиску неисправностей в вычислительных сетях.

Проект является открытым. Программное обеспечение разрабатывается и распространяется в соответствии с положениями GNU GPL.

В Net-Simulator реализованы два уровня ISO OSI: канальный и сетевой, что позволяет решать следующие задачи: изучение принципов работы коммуникаторов второго и третьего уровня, пассивных концентраторов; отработку практических навыков статической маршрутизации в IP-сетях и поиска неисправностей в IP-сетях.

Физическая природа сети не учитывается. Предполагается, что пакеты канального уровня распространяются в среде аналогичной локальной сети на основе Ethernet. На канальном уровне используется простейший Ethrnet-образный протокол, который предусматривает адресацию по 6-ти байтовым MAC адресам. Уникальность MAC адресов обеспечивает ядро Net-Simulator. Пакет канального протокола представляет собой объект Java и не имеет аналогов в реальных сетях.

На сетевом уровне используется ограниченная реализация IP в соответствии с RFC791. Для преобразования IP адресов в MAC реализована служба ARP на основе широковещательных запросов.

Для работы служебных утилит, таких как ping, используется ограниченная реализация ICMP в соответствии с RFC792.

В главном окне NET-Simulator отображается поле, в которое можно добавлять различные сетевые устройства из меню Устройства. Поддерживаются следующие типы устройств:

Маршрутизатор. Коммутатор 3-го уровня с 8-мью интерфейсами и поддержкой IP4.

Настольный компьютер. Фактически маршрутизатор с одним интерфейсом.

Концентратор (Hub). Простейшее устройства ретранслирующее пакеты канального уровня на свои интерфейсы. Не имеет терминала и соответственно никак не управляется.

Коммутатор (Switch). Коммутатор 2-го уровня с 8-мью интерфейсами. Коммутирует пакеты канального уровня на основе таблиц MAC-адресов, по аналогии с известными алгоритмами, используемыми в Ethernet-свитчах.

Устройства соединяются с помощью универсальной среды передачи данных, виртуального патчкорда. При прохождении пакета через патчкорд, он подсвечивается для визуального отслеживания активности в сети.

Вновь добавляемые устройств появляются в верхнем левом углу, после чего их можно перетаскивать мышкой в удобное место. Вилки патчкордов «приклеиваются» к розеткам интерфейсов устройств. Нажатие правой кнопки мыши на устройстве открывает контекстное меню, которое позволяет просмотреть свойства, открыть терминал или удалить устройство. Двойной щелчок левой кнопкой мыши открывает терминал.

Проекты сохраняются в формате xml. Проекты можно сохранять в виде html-отчетов. Отчет состоит непосредственно из html-файла с детальным описанием проекта и одноименного файла со схемой виртуальной сети в формате png. Отчеты формируются путем конвертации исходного xml-файла проекта

при помощи xsl-шаблона. По умолчанию используется шаблон cfg/tohtml.xsl. Изменяя шаблон можно добиться желаемого вида отчета.

Виртуальные устройства в Net-Simulator управляются при помощи интерфейса командной строки из виртуальных терминалов. Терминал устройства можно открыть двойным кликом на значке устройства или через контекстное меню. Поддерживается история команд, клавиши вверх/вниз позволяют просматривать историю команд.

Список команд доступных на данном устройстве можно посмотреть командой help. Курсивом здесь и далее выделены служебные слова и символы, набираемые на экранах виртуальных терминалов. Сочетание клавиш Ctrl+L очищает терминал. Краткая справка по любой команде выводится при вызове команды с опцией -h.

Команды Net-Simulator

В режиме командной строки пользователю доступны следующие команды: *help; route; ifconfig; ping; arp; mactable; help*.

help – выводит список доступных команд.

help [-h]

Опции Описание

-h краткая справка.

Содержимое квадратных скобок является необязательным. На позициях угловых скобок размещают соответствующие значения.

route – позволяет управлять таблицей маршрутизации устройств поддерживающих протокол IP4.

route [-h] [{-add|-del} <*target*> [-netmask <*address*>] [-gw <*address*>] [-metric <*M*>] [-dev <*If*>]]

Опции Описание

-h краткая справка.

<*target*> адрес назначения. Назначением может быть под-сеть или отдельный узел в зависимости от значения маски подсети. Если маска равна 255.255.255.255 или отсутствует совсем, назначением будет узел, иначе назначением будет сеть.

-add добавляет новый маршрут в таблицу маршрутизации.

-del удаляет маршрут из таблицы маршрутизации.

-dev <*If*> принудительно присоединяет маршрут к определенному интерфейсу. <*If*> – имя интерфейса.

-gw <*address*> направляет пакеты по этому маршруту через заданный шлюз; <*address*> – адрес шлюза.

-netmask <*address*> маска подсети используемая совместно с адресом назначения при добавлении маршрута <address> – маска. Если маска не задана явно, подразумевается 255.255.255.255.

-metric <*M*> метрика, используемая в данном маршруте. <*M*> — целое число большее или равное нулю.

Если **route** вызывается без параметров, то команда выводит на экран таблицу маршрутизации:

=>route

IP routing table

Destination	Gateway	Netmask	Flags	Metric	Iface
10.0.0.0	*	255.0.0.0	U	1	eth0
11.0.0.0	10.0.0.10	255.0.0.0	UG	1	eth0
192.168.120.1	10.0.0.10	255.255.255.255	UGH	1	eth0

Если маршрут не использует шлюз, вместо адреса шлюза выводиться *. Flags может содержать значение: U — маршрут активен, G — маршрут использует шлюз, H — назначением является узел.

Примеры:

=>route -add 192.168.120.0 -netmask 255.255.255.0 -dev eth0

=>route

IP routing table

Destination	Gateway	Netmask	Flags	Metric	Iface
192.168.120.0	*	255.255.255.0	U	1	eth0

=>

=>route -add 192.168.121.10 -gw 192.168.120.10

=>route

IP routing table

Destination	Gateway	Netmask	Flags	Metric	Iface
192.168.120.0	*	255.255.255.0	U	1	eth0
192.168.121.10	192.168.120.1	255.255.255.255	UGH	1	eth0

=>

ifconfig – конфигурирует сетевые интерфейсы.

ifconfig [-h] [-*a*] [<*interface*>] [<*address*>] [-broadcast <*address*>] [-netmask <*address*>] [-up|-down]

Опции	Описание
-h	краткая справка.
-a	показывает информацию о всех интерфейсах. Если данная опция отсутствует, выводится информация только об активных интерфейсах.
<*interface*>	конфигурирует или показывает информацию только о заданном интерфейсе.
<*address*>	IP-адрес, присваиваемый интерфейсу.
-broadcast <*address*>	широковещательный адрес, присваиваемый интерфейсу, <*address*> – широковещательный адрес.
-netmask <*address*>	маска подсети, используемая совместно с адресом; <*address*> – маска. Если маска не задана явно, маска принимается равной стандартным значения для стандартных классов подсетей A, B и C.
-up	активирует интерфейс. При активизации интер-фейса для него автоматически добавляется соответствующий маршрут в таблице маршрутизации.
-down	деактивирует интерфейс. При деактивации ин-терфейса соответствующий маршрут автоматически удаляется из таблицы маршрутизации.

Если **ifconfig** вызывается без параметров, то команда выводит на экран данные о состоянии всех активных интерфейсов:

=>ifconfig

eth0 Link encap:Ethernet HWaddr 0:0:0:0:CF:0
inet addr:192.168.120.1 Bcast:192.168.120.255 Mask:255.255.255.0
UP
RX packets:23 errors:0 dropped:0
TX packets:23 errors:0 dropped:0
RX bytes:0 TX bytes:0

HWaddr – уникальный 6-ти байтовый адрес интерфейса, ана-логичный MAC-адресу в Ethernet сетях. Назначается автоматически.

Примеры:

=>ifconfig eth0 192.168.120.1 –broadcast 192.168.120.255 –netmask 255.255.255.0 -up

=>ifconfig

eth0 Link encap:Ethernet HWaddr 0:0:0:0:CF:0
inet addr:192.168.120.1 Bcast:192.168.120.255 Mask:255.255.255.0
UP
RX packets:0 errors:0 dropped:0
TX packets:0 errors:0 dropped:0
RX bytes:0 TX bytes:0

ping – использует ICMP протокол чтобы проверить достижимость интерфейса удаленного узла. Утилита **ping** посылает удаленному узлу ICMP ECHO_REQUEST и ожидает в течении определенного промежутка времени ICMP ECHO_RESPONSE. В случае получения ответа выводит данные о прохождении ICMP-пакета по сети. (Отмена выполнения команды: «Ctrl-c»).

ping [-h] [-i <*interval*>] [-t <*ttl*>] <*destination*>

Опции Описание

-h краткая справка.

-i <*interval*> задает частоту ICMP-запросов; <*interval*> – интервал между запросами в секундах. По умолчанию отсылается один пакет в секунду.

-t <*ttl*> задает значение атрибута Time to Live в генерируемых IP-пакетах. <*ttl*> – целое число от 0 до 255. По умолчанию TTL равно 64.

<*destination*> IP-адрес исследуемого узла

Примеры:

=>ping 192.168.120.1

PING 192.168.120.1
64 bytes from 192.168.120.1: icmp_seq=0 ttl=62 time=477 ms
64 bytes from 192.168.120.1: icmp_seq=1 ttl=62 time=435 ms
64 bytes from 192.168.120.1: icmp_seq=2 ttl=62 time=234 ms
64 bytes from 192.168.120.1: icmp_seq=3 ttl=62 time=48 ms
64 bytes from 192.168.120.1: icmp_seq=4 ttl=62 time=87 ms
64 bytes from 192.168.120.1: icmp_seq=5 ttl=62 time=56 ms

ping выводит результат исследования удаленного узла в следующем формате:

64 bytes from 192.168.120.1 – размер полученного ответа и адрес источника ответа. В NET-Simulator размер пакета имеет условное значение и всегда равен 64B.

icmp_seq=0 – номер пакета. Каждый запрос содержит свой номер, как правило формируется инкрементно. ping выводит номер пакета из каждого полученного ответа.

ttl=62 – значение TTL из полученного ответа. time=48 ms — время прохождения пакетом полного маршрута (туда и обратно, round-trip time) в миллисекундах.

arp – показывает ARP-таблицу устройства. Кроме того опция -r позволяет сформировать запрос для определения MAC-адреса по явно заданному IP-адресу. Эта функция обычно отсутствует в реальных устройствах, в NET-Simulator она добавлена для наглядности при изучении протоколов канального и сетевого уровня.

arp [-h] [-r <*IP-address*> <*interface*>]

Опции	Описание
-h	краткая справка.

-r <*IP-address*> <*interface*> Прежде чем вывести ARP-таблицу, предпринимает попытку найти MAC-адрес по явно заданному IP-адресу; <*IP-address*> - IP-адрес, для которого определяется MAC-адрес. <*interface*> имя интерфейса в подсоединенной сети к которому будет происходить поиск.

Если **arp** вызывается без параметров, то команда выводит на экран ARP-таблицу:

```
=>arp
Address          HWaddress       iface
10.0.0.10        0:0:0:0:BC:0    eth0
10.0.0.11        0:0:0:0:1F:2    eth0
```

Примеры:

```
=>arp -r 192.168.120.12 eth1
=>arp
Address          HWaddress       iface
10.0.0.10        0:0:0:0:BC:0    eth0
10.0.0.11        0:0:0:0:1F:2    eth0
192.168.120.12   0:0:0:0:12:1    eth1
```

mactable – показывает таблицу MAC-адресов коммутаторов второго уровня.

mactable [-h]

Опции	Описание
-h	краткая справка.

Примеры:

```
=>mactable
MACAddress      port
0:0:0:0:B3:0    0
0:0:0:0:2F:2    2
0:0:0:0:03:0    3
```

Где port — номер порта на коммутаторе. Нумерация портов идет по порядку начиная с нуля.

Основы работы с сетевым имитатором Net-Simulator

Запуск программы Net-Simulator в среде Linux может быть выполнен из главного меню (кнопка К / Образование / Разное / Net-Simulator). При этом раскрываются два окна: окно истории функционирования симулятора и окно графического представления сети. На раскрытом рабочем окне программы Net-Simulator вкладка «Проект» содержит набор традиционных директив для работы с файлами. Вкладка «Устройства» содержит перечень сетевых компонентов, доступных для использования при моделировании сетей. Вкладка «Сервис» открывает дополнительные возможности представления анализируемого проекта. Дополнительные сведения о функционировании симулятора размещены в каталоге S:\ БогомоловСИ\ Model\ NetSim\ Net-Simulator.

Для составления новой схемы сети следует указать на пункт меню «Создать» вкладки «Проект». Сетевые компоненты (компьютеры, концентраторы, коммутаторы и т.д.), необходимые для построения схемы сети, выбирают с помощью вкладки «Устройства» с дальнейшим размещением на рабочем поле проекта.

Для соединения сетевых компонентов используют пункт меню «Среда передачи (кабель)» вкладки «Устройства». Для «подключения» кабеля к устройству следует «разъем» кабеля подтащить к изображению «гнезда» прибора. При удачном «подключении» кабеля на приборе будет подсвечен индикатор соединения соответствующего порта.

Для конфигурирования устройств следует открыть виртуальный терминал двойным «кликом» на графическом изображении соответствующего узла либо выбором пункта «Терминал» при нажатии правой кнопки мыши на выбранном устройстве. Перечень команд, доступных для имитации работы выбранного устройства, может быть получен при вводе на терминале команды *help*.

Моделирование сетевых процессов основано на имитации передачи дейтаграмм IP в сетях IEEE 802, поэтому каждому узлу сети должен быть присвоен IP адрес (как индивидуальный, так и широковещательный) и маска адреса. MAC адрес каждому узлу присваивается программой автоматически. Номера сетевых интерфейсов каждого узла начинаются с нуля, например, eth0. В устройствах, имеющих несколько портов, их нумерация на графических образах выполняется слева направо.

Для формирования отчета результаты моделирования в виде копий экрана (протоколы терминалов, схемы сетей) последовательно переносятся на графический редактор, например, Kolour-Paint {кнопка К (аналог кнопки ПУСК в среде Windows) / ГРАФИКА /}, с последующим выбором нужных сегментов экрана и переносом их в текстовый редактор, например, Write.

Предварительная подготовка

Изучить принципы функционирования сетевого уровня модели OSI, типы адресов сетей TCP/IP, классификацию IP-адресов, назначение масок в IP-адресации, ознакомиться с работой сетевого оборудования (сетевая карта, концентратор, коммутатор, маршрутизатор), с протоколами ARP и ICMP и утили-

той *ping* по материалам лекций и литературных источников, рекомендуемых, в том числе, и для самостоятельного изучения .

Ознакомиться с описанием сетевого имитатора Net-Simulator и его основных компонентов по материалам данного практикума. Изучить раздел «Основы работы с сетевым имитатором Net-Simulator» методических указаний по данной лабораторной работе и подготовить необходимые схемы моделей сетей, рассмотренные в данном разделе.

Подготовить схему ЛВС, состоящую из 3 компьютеров по технологии «звезда». В качестве IP-номера сети использовать адреса, зарезервированные для автономных локальных сетей класса С вида 192.168.XYZ.0, где XY – порядковый номер студента по списку, Z – произвольная цифра от 0 до 9. Номер узлов в сети выбирается произвольно.

Изменить схему сети, заменив концентратор коммутатором. При этом изменить номер сети одного из узлов путем изменения цифры в позиции Z.

Изменить модернизированную схему объединенной сети, вернув на место концентратор и введя маршрутизатор для логической структуризации сети. Портам маршрутизатора присвоить адреса, соответствующие используемым номерам сетей.

Дополнить схему объединенной сети введением подсетей. Для этого заменить концентратор вторым маршрутизатором, а сеть из двух компьютеров разбить на две подсети с использованием масок. Для формирования номеров подсетей использовать четыре бита младшего байта адреса. В качестве номеров подсетей использовать порядковый номер студента, и число, на единицу большее.

Результатами предварительной подготовки по данной работе являются разработанные схемы сетей с указанием IP-адресов всех сетевых компонентов.

Практическое задание

1. Ознакомиться с описанием сетевого имитатора Net-Simulator и его основных компонентов по материалам данного практикума и рекомендуемых в нем литературных источников (каталог \Lab3\Metod\). Создать папку с названием Lab_3 на сервере X. Промежуточные и окончательные результаты данной работы сохранять в этом каталоге.

2. В среде Linux запустить пакет Net-Simulator (кнопка К / Образовательные / Net-Simulator). Ознакомиться с пакетом Net-Simulator с помощью его справочной системы.

3. Выбрать пункт «Проект» главного меню рабочего окна программы Net-Simulator.; в ниспадающем подменю выбрать пункт «Создать».

4. Используя схему сети, подготовленную в результате предварительной подготовки, построить модель ЛВС, состоящую из 3 компьютеров и концентратора.

5. Открыть терминал одного из компьютеров. Проверить функционирование команд **help** и **echo**. Результаты выполнения команд сохранить в отчете.

6. Отрыть виртуальный терминал каждого компьютера и с помощью команды **ifconfig** ввести параметры сетевых интерфейсов всех узлов сети. С по-

мощью команды **route** проконтролировать таблицы маршрутизации всех компьютеров.

7. С помощью команды **arp** сформировать запросы на соседние узлы сети. С помощью команды **ifconfig** проконтролировать состояние интерфейсов всех узлов сети. Обратить внимание на количество переданных и принятых пакетов на каждом узле. Провести анализ перемещения пакетов по сети, а также количества широковещательных пакетов и пакетов с конкретным адресом назначения на основании их времени кругооборота (пакеты широковещательной рассылки удобно контролировать на тех узлах, с которыми на данный момент не проводился сеанс обмена сообщениями).

8. С помощью команды **ping** проверить достижимость любого из узлов сети. С помощью команды **ifconfig** проконтролировать состояние интерфейсов всех узлов сети. Обратить внимание на количество переданных и принятых пакетов.. Провести анализ перемещения пакетов по сети, а также количества широко-вещательных пакетов и пакетов с конкретным адресом назначения на основании их времени кругооборота.

9. Повторить п. 8 задания для всех компьютеров данной сети. Выявить закономерности формирования широковещательных пакетов.

10. Выполнить команду **ping** для проверки достижимости узла назначения сети с тем же самым адресом, что и адрес источника. Объяснить функционирование системы в этой ситуации.

11. Повторить п. 8 задания для проверки достижимости узлов сети с адресами, которые отсутствуют в данной сети. При этом в качестве адресов несуществующих узлов выбирать номера, которые могут как относиться к данной сети, так и находиться за ее пределами. Обосновать полученные результаты.

12. Увеличивая параметр *i* команды **ping** добиться, чтобы каждый отправляемый ICMP пакет требовал рассылки ARP пакетов. Оценить продолжительность хранения адресов *timeout* ARP таблиц до их удаления.

13. Используя схему, подготовленную в результате предварительной подготовки, построить модель сети, состоящую из 3 компьютеров и коммутатора.

14. Открыть терминал одного из двух компьютеров, принадлежащих к одной сети. С помощью команды **ping** проверить достижимость второго узла сети. С помощью команды **ifconfig** проконтролировать состояние интерфейсов всех узлов сети.

15. С помощью команды **ping** проверить достижимость компьютера, находящегося в другой сети. С помощью команды **ifconfig** проконтролировать состояние интерфейсов всех узлов сети. Провести анализ перемещаемых пакетов. Объяснить полученные результаты.

16. С помощью команды **route** ввести в таблицу маршрутизации информацию о маршруте, обеспечивающем передачу пакетов во вторую сеть.

17. Повторить п. 14 задания. С помощью команды **route** ввести в таблицу маршрутизации компьютера-получателя информацию о маршруте, обеспечивающем передачу пакетов во вторую (для данного компьютера) сеть. Снова повторить п. 14. Объяснить полученные результаты.

18. Используя схему, подготовленную в результате предварительной подготовки, построить модель сетей, объединенных с помощью маршрутизатора.

19. Повторить п.п. 13…16 для одного из двух компьютеров, размещенных в одной сети.

20. Используя схему, подготовленную в результате предварительной подготовки, построить модель объединенной сети, с разбиением на подсети помощью маршрутизатора.

21. С помощью команды **ifconfig** ввести параметры сетевых интерфейсов всех узлов сети. С помощью команды **route** вести параметры таблицы маршрутизации, обеспечивающих достижимость любого узла сети с любого компьютера.

22. С помощью команды **ping** проверить достижимость всех узлов спроектированной сети для всех компьютеров.

Контрольные вопросы и задания

1. Какие типы адресов используются в сетях TCP/IP?
2. Что такое локальный адрес и как он используется?
3. Назначение IP-адреса.
4. С какой целью введены символьные имена узлов сети?
5. Каким образом и для чего IP-адреса разбиты на классы?
6. Что такое группой адрес и как он используется?
7. Что такое широковещательная рассылка?
8. Какая информация содержится в адресном поле IP пакета для выполнения широковещательной рассылки в удаленной сети?
9. Какая информация содержится в адресном поле IP пакета для выполнения широковещательной рассылки в локальной сети?
10. Какие ограничения накладываются на выбор IP-адресов?
11. Что понимается под термином loopback?
12. Для каких целей используется адрес сети 127.0.0.0?
13. Какую функцию выполняет маска адреса?
14. Указать маску для сетей класса A (в двоичном и десятичном формате).
15. Привести маску для сетей класса B (в двоичном и десятичном формате).
16. Привести маску для сетей класса C.
17. Указать маску для сетей класса C (в двоичном и десятичном формате).
18. Что следует понимать под адресами класса D?
19. Что такое таблица маршрутизации?
20. Основные компоненты таблицы маршрутизации.
21. Какие сетевые компоненты используют таблицу маршрутизации?
22. Какие сетевые компоненты не используют таблицу маршрутизации?
23. Что представляет собой проект Net-Simulator?
24. Какие компоненты сети могут быть смоделированы в Net-Simulator?
25. Какие команды используются в Net-Simulator?
26. Какие команды используются для контроля конфигурации сетевых интерфейсов?

27. Какие команды используются для контроля таблицы маршрутизации?
28. Для каких целей используется команда *ping*?
29. Какие протоколы используются при применении команды *ping*?

4 РЕКОМЕНДАЦИИ К ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Федеральный государственный образовательный стандарт определил, что выпускник по направлению подготовки «Инфокоммуникационные технологии и системы связи» с квалификацией (степенью) «магистр» в соответствии с задачами профессиональной деятельности и целями основной образовательной программы должен обладать определенным набором общекультурных (ОК) и профессиональных (ПК) компетенций. Причем ряд этих компетенций ориентирован на самостоятельное развитие профессионального и общекультурного уровня:

способен совершенствовать и развивать свой интеллектуальный и общекультурный уровень (ОК-1);

способен к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности (ОК-2);

способен самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности (ОК-6);

способен самостоятельно выполнять экспериментальные исследования для решения научно-исследовательских и производственных задач с использованием современной аппаратуры и методов исследования; способен участвовать в научных исследованиях в группе, ставить задачи исследования, выбирать методы экспериментальной работы (ПК-9).

При изучении данной дисциплины так же как и при изучении других дисциплин данного направления подготовки значительный ресурс трудоемкости ориентирован на самостоятельную работу студентов. В данном разделе указаны основные направления самостоятельной работы студентов, используемые при изучении данной дисциплины.

При подготовке к практическим занятиям рекомендуется использовать соответствующие разделы лекций и материалов, указанных в перечне литературы. Приветствуется использование дополнительных источников. Особое внимание следует обратить на вопросы, перечисленные во вводной части каждого практического занятия.

Организация самостоятельной работы при подготовке к лабораторным занятиям рассмотрена в соответствующем разделе методического пособия по проведению лабораторных работ по данной дисциплине.

Вопросы для самопроверки и задания при подготовке к лабораторным работам приведены ниже.

Исследование основных компонентов сетевого имитатора NS2

1. С какой целью используется динамическое моделирование систем? Какие задачи оно решает?
2. Дать сравнительную характеристику программных продуктов, предназначенных для моделирования телекоммуникационных сетей.
3. Что представляет собой проект NS2 VINT?
4. Раскрыть архитектуру имитатора NS2.
5. С какой целью в NS2 используется два языка программирования?
6. Как в NS2 отражены реальные характеристики сетевых протоколов, порядок обслуживания очередей?
7. Какие виды ошибок могут быть смоделированы в NS2?
8. Какие средства используются для визуализации в NS2?
9. Какие основные компоненты содержит шаблон Tcl сценария?
10. Пояснить элементы Tcl сценариев, необходимые при создании узлов и связей между ними.
11. Перечислить основные параметры линии связи между узлами и пояснить, как они задаются в Tcl сценарии.
12. Каким образом можно вручную размещать компоненты на схеме сети? Привести примеры.
13. Что такое агенты и какие функции они выполняют?
14. Перечислить известные агенты и описать их основные характеристики.
15. Какие параметры агентов учитываются при моделировании сети и каким образом?
16. Что такое CBR генераторы и как они участвуют в моделировании?
17. Какие параметры CBR генераторов могут быть заданы при моделировании?
18. Что такое планирование событий и как оно реализуется в NS2?
19. Каким образом можно контролировать потоки данных?
20. Для чего и как выполняется маркировка данных?
21. Какие виды организации очереди используются в NS2? Привести примеры.
22. Какие параметры пакета могут быть определены в результате эксперимента?
23. Какими средствами в NS2 отображаются результаты эксперимента? Пояснить.

Моделирование сетей ЭВМ с помощью сетевого имитатора NS2

1. С чего начинается NS сценарий?
2. Каковы результаты действия первой строки NS сценария?
3. Каковы возможности методов объекта Simulator?
4. Как создаются узлы сети в NS симуляторе? Пример программы.
5. На каком уровне стека ЭМВОС (TCP/IP) работают узлы в NS? По каким признакам различаются узлы на этом уровне?
6. Как создаются в NS соединения между узлами?

7. Какие параметры соединения устанавливаются при моделировании сети?

8. На каком уровне стека ЭМВОС (TCP/IP) работают линии связи в NS? По каким признакам различаются работают линии связи на этом уровне?

9. Что такое агенты? Какие параметры агентов устанавливаются при моделировании сети?

10. Как агенты связаны с узлами сети и между собой?

11. На каком уровне стека ЭМВОС (TCP/IP) работают агенты в NS? По каким признакам различаются агенты на этом уровне?

12. На каком уровне стека ЭМВОС (TCP/IP) работают источники трафика в NS? По каким признакам различаются работают источники трафика на этом уровне?

13. В каких случаях пакет попадает в очередь? Каким образом он ее покидает?

14. Какие параметры очереди устанавливаются при моделировании соединения?

15. При каких условиях пакет удаляется из сети?

16. Что такое планирование событий? Как оно реализуется в NS?

17. С какой целью в NS используется файл трассировки?

18. Раскрыть формат данных в файле трассировки.

19. Пояснить условные обозначения данных файла трассировки.

20. Что такое флаги в заголовке пакета сетевого уровня? С какой целью они используются?

21. Каким образом в NS моделируются аварийные ситуации в сети?

22. Что происходит с пакетом UDP при аварийной ситуации в сети?

23. Что происходит с пакетом TCP при аварийной ситуации в сети?

24. Что такое динамическая маршрутизация сети?

25. Как динамическая маршрутизация реализуется в NS?

Исследование характеристик протокола TCP с помощью сетевого имитатора NS2

1. Какие задачи решает протокол TCP?

2. Сопоставить стек TCP и стек OSI.

3. Какие механизмы использует протокол TCP для надежной доставки данных?

4. Какая информация передается в полях «номера портов» и «номера последовательностей» заголовка TCP сегмента?

5. Какая информация передается в полях «флаги» заголовка TCP сегмента?

6. Какая информация передается в полях «размер окна» и «контрольная сумма» заголовка TCP сегмента?

7. Порядок установления TCP соединения.

8. Как завершается TCP соединение в штатном режиме?

9. Как завершается TCP соединение в особых случаях?

10. Какие состояния можно выделить в процессе TCP соединения?

11. Особенности работы TCP с интерактивными данными.
12. В чем заключается алгоритм Нейгла?
13. Особенности передачи TCP большого объема данных.
14. Особенности реализации алгоритма «скользящее окно» в протоколе TCP.
15. Пояснить механизм «скользящего окна».
16. Из каких соображений выбирается размер окна?
17. В каких случаях в заголовке пакета устанавливается флаг «PUSH»? Как на это реагирует получатель?
18. В чем заключается алгоритм медленного старта?
19. С какой целью и как используется параметр «окно переполнения»?
20. Раскрыть понятие «сокет». В каких полях заголовка содержится информация о нем?
21. В чем заключается квитирование при передаче данных?
22. Особенности квитирования в протоколе TCP.
23. В чем особенности модели агента TCP в симуляторе NS?
24. С какой целью и как определяется время кругооборота?
25. Для какой цели и как в протоколе TCP используются таймеры?

Изучение теоретического материала целесообразно сопровождать знакомством с многочисленной справочной и нормативной документацией: протоколами, рекомендациями и стандартами. Следует иметь ввиду, что значительная доля свежей информации может быть представлена на иностранных языках. В соответствии с Государственным образовательным стандартом

При подготовке к экзаменам изучение теоретического материала рекомендуется ориентироваться на примерный перечень вопросов, выносимых на экзамен, который приведен в разделе 5 настоящего пособия.

5 ПРИМЕРНЫЙ ПЕРЕЧЕНЬ ВОПРОСОВ, ВЫНОСИМЫХ НА ЭКЗАМЕН

1. Основные сетевые стандарты
2. Подуровень управления логическим каналом (802.2)
3. Технология Ethernet (802.3). Форматы кадров технологии Ethernet
4. Протоколы и стандарты локальных сетей
5. Стандарты Ethernet (802.3)
6. Token Ring (802.5). Технология FDDI
7. Распределенная обработка информации в системах клиент-сервер
8. Механизмы взаимодействия процессов в сетях
9. Модель клиент-сервер на базе микроядра. Режим пользователя
10. Модель клиент-сервер. Режим ядра
11. Одноранговые сети
12. Интеграция локальных сетей в региональные и глобальные сети
13. Принципы маршрутизации
14. Протоколы маршрутизации
15. Типы адресов стека TCP/IP
16. Использование масок в IP адресации. Распределение IP-адресов

17. Отображение IP-адресов на локальные адреса
18. Отображение доменных имен на IP-адреса
19. Система доменных имен DNS
20. Протокол IP. Структура IP пакета
21. Маршрутизация в IP сетях
22. Использование масок в IP сетях. Фрагментация IP пакетов
23. Протокол TCP
24. Реализация скользящего окна в протоколе TCP
25. Внутренние и внешние протоколы маршрутизации в IP сетях. Протокол OSPF
26. Дистанционно-векторный протокол RIP
27. Неоднородные вычислительные сети. Шлюзы
28. Мультиплексирование стеков протоколов
29. Основные понятия безопасности. Шифрование
30. Технологии аутентификации. Аутентификация информации
31. Аутентификация на основе сертификатов
32. Классификация операционных систем
33. Структура операционных систем. Взаимодействие сетевых компонентов
34. ОС UNIX. Основные протоколы, службы. Архитектура ОС
35. Подсистемы ядра ОС Unix. Функционирование системы
36. Сетевая ОС Novell Netware. Основные протоколы, службы.

6 ЛИТЕРАТУРА

6.1 Основная литература

1.1 Олифер В.Г., Олифер Н.А. Компьютерные сети. С.-Петербург, изд-во «Питер».2007. - 957с. [40 экз.]

1.2 Олифер В.Г., Олифер Н.А. Сетевые операционные системы. С.-Петербург, изд-во «Питер».2007. - 538с. [10 экз.]

6.2 Дополнительная литература

2.1 Таненбаум Э. Компьютерные сети С.-Петербург, изд-во "Питер", 2002.-846с. [3 экз.]

2.2 Пуговкин А.В. Телекоммуникационные системы. – Томск: ТУСУР, 2007. - 201с. [191 экз.]

2.3 Семенов Ю.А. «Сети Интернет. Архитектура и протоко-лы» Москва, издательство «Блик плюс», 1998.- 424с. [4 экз.]

2.4. Д.В. Иртегов. Введение в сетевые технологии. Учебное пособие для вузов. СПб.: БХВ-Петербург, 2004.-539 с.

2.5 О.Р. Лапонина. Протокол SSH. Интернет ресурс http://www.intuit.ru/department/security/networksec2/10/.

2.6. Д.Н. Колисниченко, Питер В. Аллен. Linux. Полное руководство.- СПб: Наука и техника, 2006.- 784 с.

2.7. С.Л. Скловская. Команды Linux. Справочник. СПб: ООО"ДиаСофтЮП", 2004. - 848 с.

Электронный ресурс:

2.8 http://qucs.sourceforge.net

2.9 http://www.isi.edu/.

2.10 http://www.isi.edu/, http://wwwns2.chat.ru/

2.11 http://www.isi.edu/nsnam/ns/ns-documentation.html

2.12 http://www.msen.com/~clif/TclTutor.html.

2.13 http://sourceforge.net/projects/net-simulator/

Учебно-методическое пособие

Богомолов С.И.

Сети ЭВМ и телекоммуникации

Методические указания
по проведению практических занятий и
организации самостоятельной работы студентов,
обучающихся по направлению подготовки магистров 210700
«Инфокоммуникационные технологии и системы связи»

Усл. печ. л._____ Препринт
Томский государственный университет
систем управления и радиоэлектроники
634050, г.Томск, пр.Ленина, 40