МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ (МАДИ)»
ВОЛЖСКИЙ ФИЛИАЛ

Кафедра гуманитарных и естественнонаучных дисциплин

Методические указания к лабораторным работам
по дисциплине

СЕТИ ЭВМ И ТЕЛЕКОММУНИКАЦИИ

Направление подготовки

09.03.01 Информатика и вычислительная техника

Направленность (профиль, специализация) образовательной программы

«Автоматизированные системы обработки информации и управления»

Квалификация

бакалавр

Чебоксары
2019

СОДЕРЖАНИЕ

ЛАБОРАТОРНАЯ РАБОТА №1.

Знакомство с симулятором Cisco Packet Tracer 4.0.

Цель: знакомство с симулятором Cisco Packet Trace 4.0 и получение базовых навыков по работе с ним.

1. Основные сведения

Симулятор **Cisco Packet Tracer** позволяет проектировать свои собственные сети, создавая и отправляя различные пакеты данных, сохранять и комментировать свою работу.

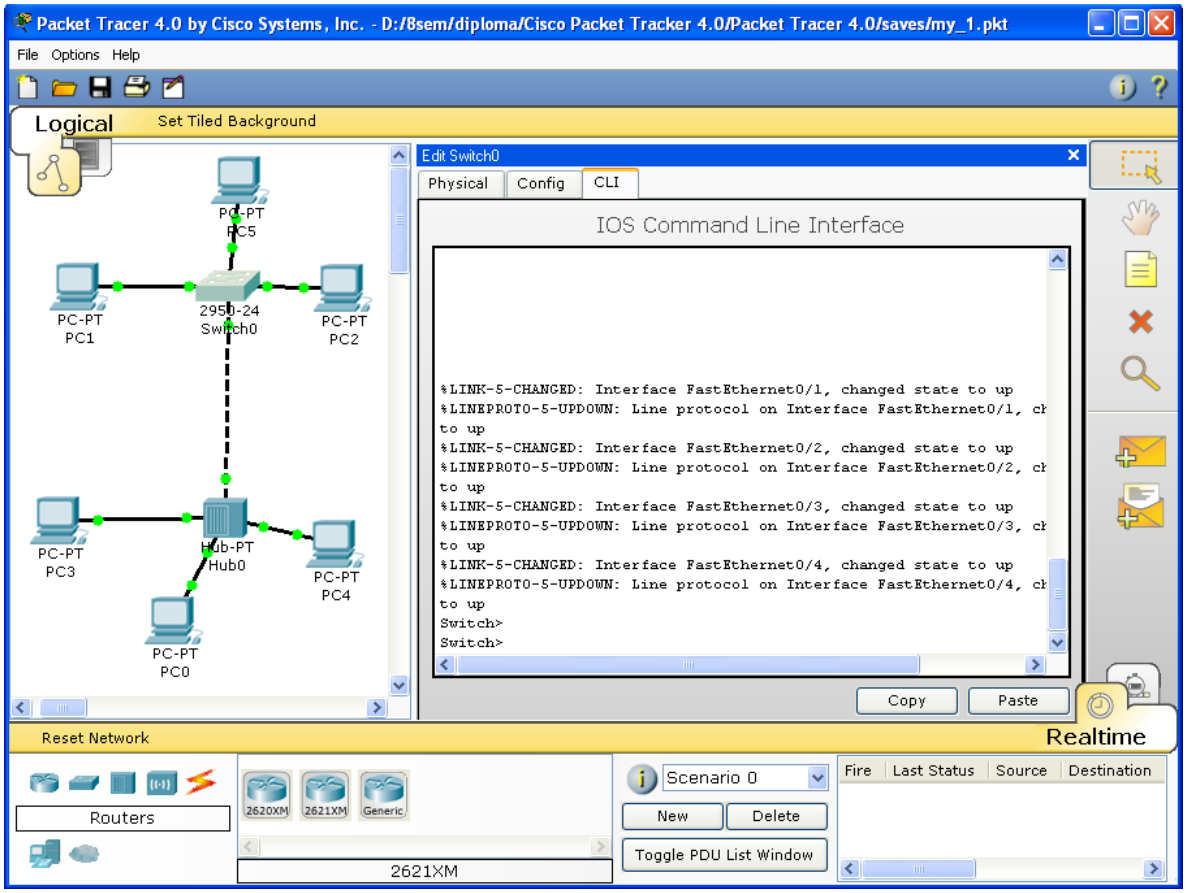


Рис.1.1 Cisco Packet Tracer 4.0

Отличительной особенностью данного симулятора является наличие в нем «Режима симуляции» (рис.1.2). В данном режиме все пакеты, пересылаемые внутри сети, отображаются графически. Эта возможность позволяет студентам наглядно продемонстрировать, по какому интерфейсу в данные момент перемещается пакет, какой протокол используется и т.д.

Однако, это не все преимущества Packet Tracer: в «Режиме симуляции» студент может не только отслеживать используемые протоколы, но и видеть, на каком из семи уровней модели OSI данный протокол задействован (см.рис.1.3).

Такая кажущаяся на первый взгляд простота и наглядность делает практические занятия чрезвычайно полезными, совмещая в них как получение, так и закрепление полученного материала.

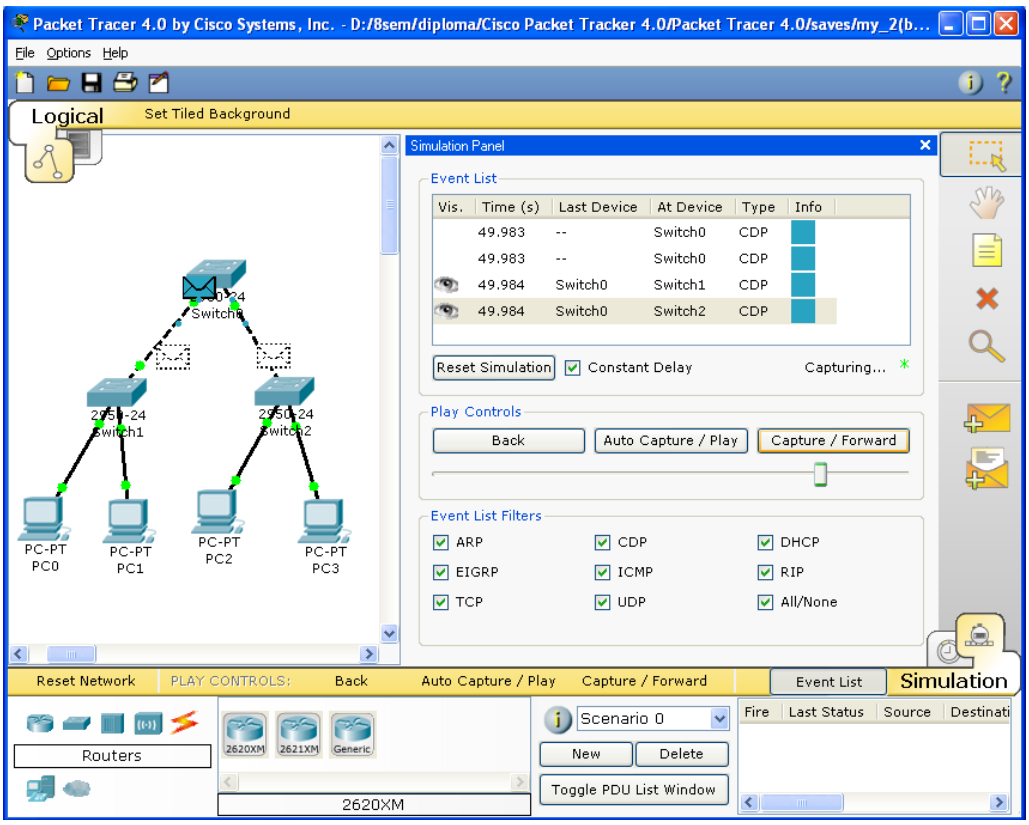


Рис.1.2 Режим «Симуляции» в Cisco Packet Tracer 4.0

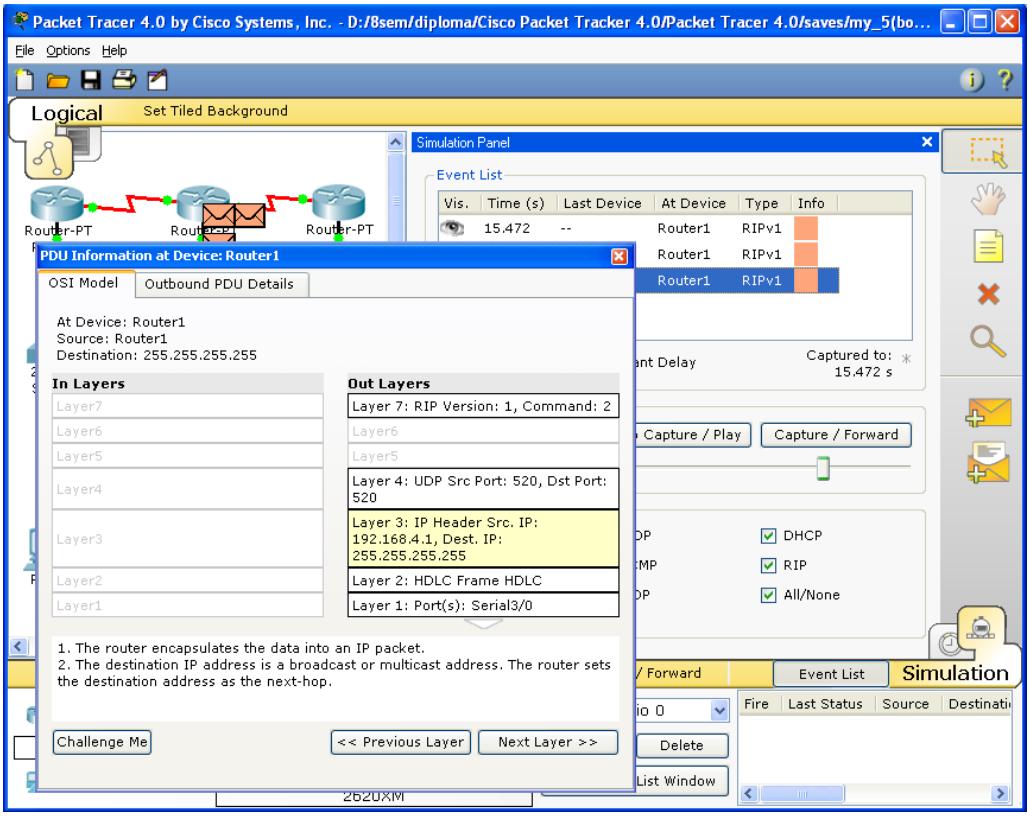


Рис.1.3 Анализ семиуровневой модели OSI в Cisco Packet Tracer 4.0

Packet Tracer способен моделировать большое количество устройств различного назначения, а так же немало различных типов связей, что позволяет проектировать сети любого размера на высоком уровне сложности:

моделируемые устройства:

коммутаторы третьего уровня:

Router 2620 XM;

Router 2621 XM;
Router-PT.
Коммутаторы второго уровня:
Switch 2950-24;
Switch 2950T;
Switch-PT;
соединение типа «мост» Bridge-PT.
Сетевые концентраторы:
Hub-PT;
повторитель Repeater-PT.
Оконечные устройства:
рабочая станция PC-PT;
сервер Server-PT;
принтер Printer-PT.
Беспроводные устройства:
точка доступа AccessPoint-PT.
Глобальная сеть WAN.
Типы связей:
консоль;
медный кабель без перекрещивания (прямой кабель);
медный кабель с перекрещиванием (кросс-кабель);
волоконно-оптический кабель;
телефонная линия;
Serial DCE;
Serial DTE.
Так же целесообразно привести те протоколы, которые студент может отслеживать:
ARP;
CDP;
DHCP;
EIGRP;
ICMP;
RIP;
TCP;
UDP.

Описание терминального режима

Маршрутизатор конфигурируется в командной строке операционной системы Cisco IOS. Подсоединение к маршрутизатору осуществляется через Telnet на IP-адрес любого из его интерфейсов или с помощью любой терминальной программы через последовательный порт компьютера, связанный с консольным портом маршрутизатора. Последний способ предпочтительнее, потому что процесс конфигурирования маршрутизатора может изменять параметры IP-интерфейсов, что приведет к потере соединения, установленного через Telnet. Кроме того, по соображениям безопасности доступ к маршрутизатору через Telnet следует запретить [8].

В рамках данного курса конфигурация маршрутизаторов будет осуществляться посредством терминала.

При работе в командной строке Cisco IOS существует несколько контекстов (режимов ввода команд).

Контекст пользователя открывается при подсоединении к маршрутизатору; обычно при подключении через сеть требуется пароль, а при подключении через консольный порт пароль не нужен. В этот же контекст командная строка автоматически переходит при продолжительном отсутствии ввода в контексте администратора. В контексте пользователя доступны только простые команды (некоторые базовые операции для мониторинга), не влияющие на конфигурацию маршрутизатора. Вид приглашения командной строки:

router>

Вместо слова router выводится имя маршрутизатора, если оно установлено.

Контекст администратора (контекст "exec") открывается командой **enable**, поданной в контексте пользователя; при этом обычно требуется пароль администратора. В контексте администратора доступны команды, позволяющие получить полную информацию о конфигурации маршрутизатора и его состоянии, команды перехода в режим конфигурирования, команды сохранения и загрузки конфигурации. Вид приглашения командной строки:

router#

Обратный переход в контекст пользователя производится по команде **disable** или по истечении установленного времени неактивности. Завершение сеанса работы - команда **exit.**

Глобальный контекст конфигурирования открывается командой **config terminal** ("конфигурировать через терминал"), поданной в контексте администратора. Глобальный контекст конфигурирования содержит как непосредственно команды конфигурирования маршрутизатора, так и команды перехода в контексты конфигурирования подсистем маршрутизатора, например:

контекст конфигурирования интерфейса

открывается командой **interface *имя_интерфейса*** (например **interface serial0**), поданной в глобальном контексте конфигурирования;

контекст конфигурирования процесса динамической маршрутизации

открывается командой **router *протокол номер_процесса*** (например, **router ospf 1**, поданной в глобальном контексте конфигурирования.

Существует множество других контекстов конфигурирования. Некоторые контексты конфигурирования находятся внтури других контекстов конфигурирования.

Вид приглашения командной строки в контекстах конфигурирования, которые будут всречаться наиболее часто:

router(config)# /глобальный/
router(config-if)# /интерфейса/
rounter(config-router)# /динамической маршрутизации/
rounter(config-line)# /терминальной линии/

ВАЖНО! Студенты должны запомнить вид приглашений командой строки во всех вышеуказанных контекстах и правила перехода из контекста в контекст. В дальнейшем примеры команд всегда будут даваться вместе с приглашениями, из которых студенты должны определять контекст, в котором подается команда. Примеры не будут содержать указаний, как попасть в необходимый контекст.

Выход из глобального контекста конфигурирования в контекст администратора, а также выход из любого подконтекста конфигурирования в контекст верхнего уровня производится командой **exit** или **Ctrl-Z**. Кроме того, команда **end**, поданная в любом из контекстов конфигурирования немедленно завершает процесс конфигурирования и возвращает оператора в контекст администратора.

ВАЖНО! Любая команда конфигурации вступает в действие немедленно после ввода, а не после возврата в контекст администратора.

Упрощенная схема контекстов представлена на рис.1.4.

Все команды и параметры могут быть сокращены (например, "**enable**" - "**en**", "**configure terminal**" - "**conf t**"); если сокращение окажется неоднозначным, маршрутизатор сообщит об этом, а по нажатию табуляции выдаст варианты, соответствующие введенному фрагменту [2].

В любом месте командной строки для получения помощи может быть использован вопросительный знак:

router#**?** /список всех команд данного контекста с
комментариями/

router#**co?** /список всех слов в этом контексте ввода,
начинающихся на "co" - нет пробела перед "?"/

router#**conf ?** /список всех параметров, которые могут
следовать за командой config - перед "?" есть пробел/

Подробнее о конфигурировании устройств можно узнать из [1], [2], [8].

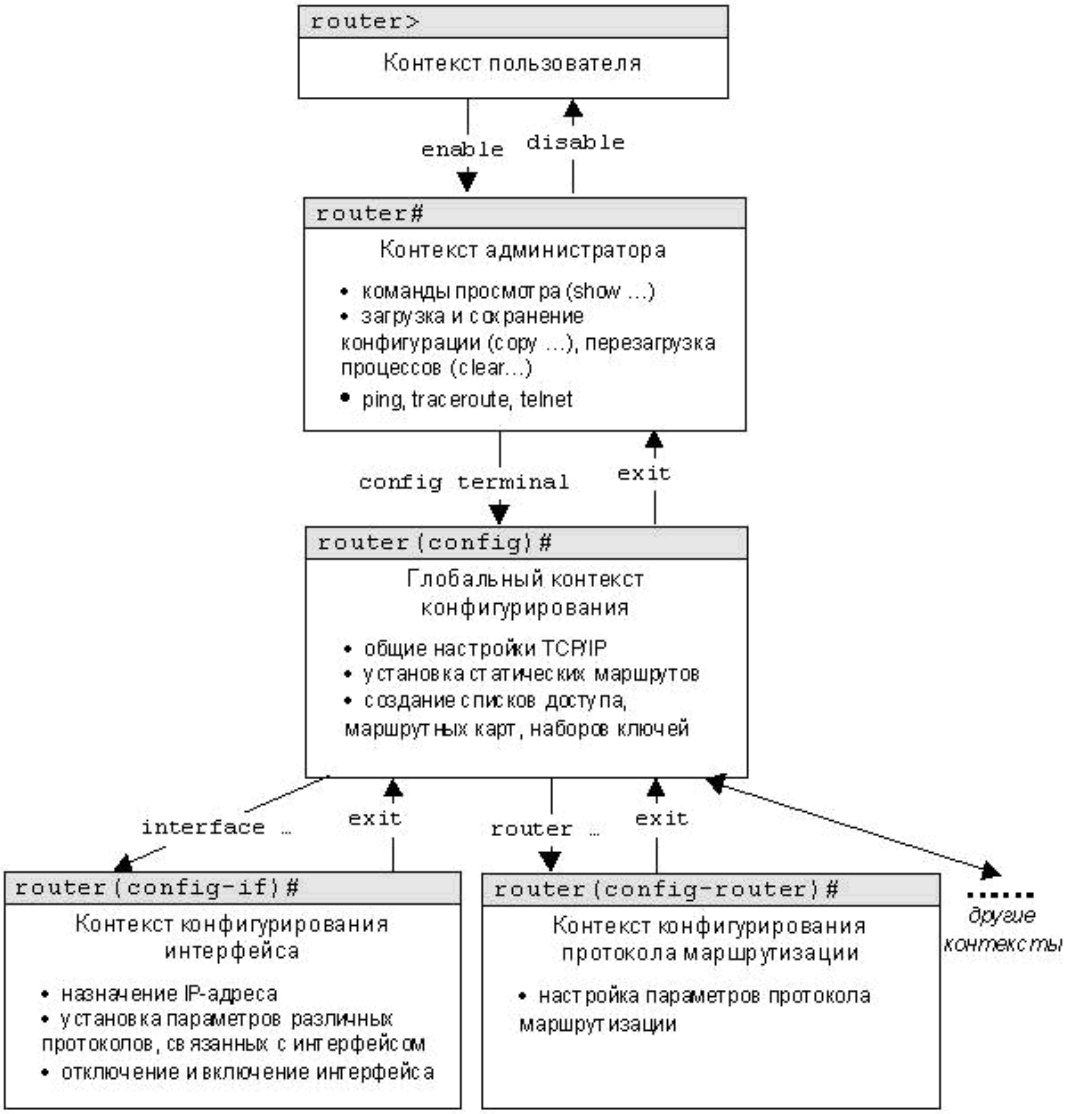


Рис.1.4. Схема контекстов Cisco IOS

Список команд

Данный список команд сгруппирован в соответствии с контекстами, в котором они [команды] применяются. В данном списке собраны те команды конфигурирования, которые необходимы для выполнения всех лабораторных работ.

Глобальный контекст конфигурирования. Команда «Access-list»

Критерии фильтрации задаются в списке операторов разрешения и запрета, называемом списком доступа. Строки списка доступа сравниваются с IP-адресами и другой информацией пакета данных последовательно в том порядке, в котором были заданы, пока не будет найдено совпадение. При совпадении осуществляется выход из списка. При этом работа списка доступа напрямую зависит от порядка следования строк.

Списки доступа имеют 2 *правила*: permit – разрешить, и deny – запретить. Именно они определяют, пропустить пакет дальше или запретить ему доступ.

Списки доступа бывают 2-ух типов: standard – стандартные (номера с 1 до 99) и extended – расширенные (номера с 100 до 199). Различия заключаются в возможности фильтровать пакеты не только по ip-адресу, но и по другим параметрам.

Формат команды (стандартные списки доступа):

access-list *номер_списка/имя правило A.B.C.D a.b.c.d* , где A.B.C.D a.b.c.d – ip-адрес и подстановочная маска соответственно.

Пример выполнения команды:

```
Router(config)#access-list 10 deny 192.168.3.0 0.0.0.3
Router(config)#
```

Данная команда означает, что данный список доступа блокирует любые пакеты с ip-адресами 192.168.3.1 - 192.168.3.3.

Команда «Enable secret»

Обычно при входе в привилегированный режим требуется ввести пароль. Данная функция позволяет предотвратить несанкционированный доступ в данный режим, ведь именно из него можно изменять конфигурацию устройства. Данная команда позволяет установить такой пароль.

Формат команды:

enable secret *пароль*

Пример выполнения команды:

После того, как был установлен пароль, при попытке входа в привилегированный режим, коммутатор будет требовать от пользователя его ввести – в противном случае вход будет невозможен.

Команда «Interface»

Команда для входа в режим конфигурирования интерфейсов конфигурируемого устройства. Данный режим представляет собой одно из подмножеств режима глобального конфигурирования и позволяет настраивать один из доступных сетевых интерфейсов (fa 0/0, s 2/0 и т.д.). Все изменения, вносимые в конфигурацию коммутатора в данном режиме относятся только к выбранному интерфейсу.

Формат команды (возможны 3 варианта):

```
Switch(config)#enable secret 123
Switch(config)#
%SYS-5-CONFIG_I: Configured from console by console
Switch#exit
Switch con0 is now available
Press RETURN to get started.
Switch>enable
Password:
Switch#
```

interface *тип порт*
interface *тип слот/порт*
interface *тип слот/подслот/порт*

Примеры выполнения команды:

После введения данной команды с указанным интерфейсом пользователь имеет возможность приступить к его конфигурированию. Необходимо заметить, что, находясь в режиме конфигурирования интерфейса, вид приглашения командной строки не отображает имя данного интерфейса.

Команда «Ip route»

```
Router(config)#ip route 76.115.253.0 255.0.0.0 76.115.252.0
Router(config)#
Router(config)#ip route 0.0.0.0 0.0.0.0 Serial2/0
Router(config)#
```

Статическая маршрутизация предполагает фиксированную структуру сети: каждый маршрутизатор в сети точно знает, куда нужно отправлять пакет, чтобы он был доставлен по назначению. Для этого можно прописать статические маршруты, используя данную команду. Команда может быть записана в двух форматах:

Первый формат команды:

ip route *A.B.C.D a.b.c.d A1.B1.C1.D1 ,*

где A.B.C.D и a.b.c.d – сетевой адрес и маска подсети, куда необходимо доставить пакеты, A1.B1.C1.D1 – ip-адрес следующего маршрутизатора в пути или адрес сети другого маршрутизатора из таблицы маршрутизации, куда должны переадресовываться пакеты;

Второй формат команды:

ip route *A.B.C.D a.b.c.d выходной_интерфейс_текущего_маршрутизатора*

Примеры выполнения команды:

Данной командой указывается маршрут, по которому пакеты из одной подсети будут доставляться в другую. Маршрут по умолчанию (Router(config)#ip route 0.0.0.0 0.0.0.0 serial 2/0) указывает, что пакеты, предназначенные узлам в другой подсети должны отправляться через данный шлюз.

```
Switch(config)#interface vlan 1
Switch(config-if)#

Router(config)#interface s 3/0
Router(config-if)#
```

Команда «Hostname»

Данная команда используется для изменения имени конфигурируемого устройства.
Формат команды:
hostname *новое_имя*
Пример выполнения команды:

Как видно, маршрутизатор поменял своё имя с Router на R1.

```
Router(config)#hostname R1
R1(config)#
```

Команда «Router rip»

RIP – Routing Information Protocol – протокол динамической маршрутизации. При его использовании отпадает необходимость вручную прописывать все маршруты – необходимо лишь указать адреса сетей, с которыми нужно обмениваться данными. Данная команда позволяет включить rip-протокол.

Пример выполнения команды:

```
Router(config)#router rip
Router(config-router)#
```

Данная команда включает rip-протокол на данном маршрутизаторе. Дальнейшая настройка производится из соответствующего контекста маршрутизации, описанного отдельно.

Контекст конфигурирования интерфейса

Команда «Ip access-group»

Данная команда используется для наложения списков доступа. Список накладывается на конкретный интерфейс, и указывается один из 2-ух параметров: in (на входящие пакеты) или out (на исходящие). Необходимо знать, что на каждом интерфейсе может быть включен только один список доступа.
Формат команды:
ip access-group *номер_списка/имя_параметр*

Пример выполнения команды:

```
Router(config-if)# ip access group 10 in
Router(config-if)#
```

В данном примере да выбранный интерфейс накладывается список доступа под номером 10: он будет проверять все входящие в интерфейс пакеты, так как выбран параметр in.

Команда «Bandwidth»

Данная команда используется только в последовательных интерфейсах и служит для установки ширины полосы пропускания. Значение устанавливается в килобитах.

Формат команды:

bandwidth *ширина_полосы_пропускания*

Пример выполнения команды:

```
Router(config)#interface serial 2/0
Router(config-if)#bandwidth 560
Router(config-if)#
```

После выполнения данной команды ширина полосы пропускания для serial 2/0 будет равна 560 kbits.

Команда «Clock rate»

Для корректной работы участка сети, где используется последовательный сетевой интерфейс, один из коммутаторов 3-его уровня должен предоставлять тактовую частоту. Это может быть оконечное кабельное устройство DCE (расшифровать). Так как маршрутизаторы CISCO являются по умолчанию устройствами DTE, то необходимо явно указать интерфейсу на предоставление тактовой частоты, если этот интерфейс работает в режиме DCE. Для этого используют данную команду (значение устанавливается в битах в секунду).

Формат команды:

clock rate *татовая_частота*

Пример выполнения команды:

```
Router(config)#interface serial 2/0
Router(config-if)#clock rate 56000
Router(config-if)#
```

После выполнения данной команды тактовая частота для serial 2/0 будет равна 56000 bits per second.

Команда «Ip address»

Каждый интерфейс должен обладать своим уникальным ip-адресом – иначе взаимодействие устройств по данному интерфейсу не сможет быть осуществлено. Данная команда используется для задания ip-адреса выбранному интерфейсу.

Формат команды:

ip address *A.B.C.D a.b.c.d* ,

где A.B.C.D a.b.c.d – ip-адрес и маска подсети соответственно.

Пример выполнения команды:

```
Switch(config)#interface vlan 1
Switch(config-if)#ip address 172.16.10.5 255.255.0.0
Switch(config-if)#
```

Результат можно проверить командой

Switch#show ip interface vlan 1

Данной командой интерфейсу vlan 1 назначен ip-адрес 172.16.10.5 с маской подсети 255.255.0.0.

Команда «No»

Данная команда применяется в случае необходимости отменить действие какой-либо команды конфигурирования.

Формат команды:

no *команда_которую_следует_отменить*

Пример выполнения команды:

```
Switch(config-if)# no shutdown
%LINK-5-CHANGED: Interface Vlan1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state to up
Switch(config-if)#
```

В данном примере использовалась команда shutdown, которая отключает выбранный интерфейс. В итоге после выполнения no shutdown интерфейс включается.

Контекст администратора

Команда «Configure terminal»

Для конфигурирования устройства, работающего под управлением IOS, следует использовать привилегированную команду configure. Эта команда переводит контекст пользователя в так называемый «режим глобальной конфигурации» и имеет три варианта:

- конфигурирование с терминала;
- конфигурирование из памяти;
- конфигурирование через сеть.

В рамках данного лабораторного курса конфигурирование будет производиться **только** посредством терминала.

Из режима глобальной конфигурации можно делать изменения, который касаются устройства в целом. Также данный режим позволяет входить в режим конфигурирования определенного интерфейса.

Пример выполнения команды:

```
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

Переход в режим глобальной конфигурации, о чем свидетельствует изменившийся вид приглашения командной строки.

Команда «Copy»

После настройки коммутатора рекомендуется сохранять его текущую конфигурацию. Информация помещается в энергонезависимую память и хранится там столько, сколько нужно. При необходимости все настройки могу быть восстановлены или сброшены.

Формат команды:

copy *running-config startup-config* – команда для сохранения конфигурации

copy *startup-config running-config* – команда для загрузки конфигурации

Пример выполнения команды:

```
Switch#copy running-config startup-config
Building configuration...
[OK]
Switch#
```

В данном примере текущая конфигурация коммутатора была сохранена в энергонезависимую память.

Команда «Show»

Show (англ. - показывать) – одна из наиболее важных команд, использующихся при настройке коммутаторов. Она применяется для просмотра информации любого рода и применяется практически во всех контекстах. Эта команда имеет больше всех параметров.

Здесь будут рассмотрены только те параметры, которые требуются в рамках данного курса. Другие параметры студент может изучить самостоятельно.

Параметр «running-config» команды «Show»

Для просмотра текущей работающей конфигурации коммутатора используется данная команда.

Пример выполнения команды:

```
Switch#show running-config
!
version 12.1
!
hostname Switch
...
```

На экран выводится текущие настройки коммутатора.

Параметр «startup-config» команды «Show»

Для просмотра сохраненной конфигурации используется данная команда.

Пример выполнения команды:

```
Switch#show startup-config
Using 1540 bytes
!
version 12.1
!
...
```

Если энергонезависим память не содержит информации, тогда коммутатор выдаст сообщение о том, что конфигурация не была сохранена.

Пример выполнения команды:

```
Switch #show startup-config
startup-config is not present
Switch #
```

Вывод сообщения о том, что в памяти отсутствует какая-либо информация.

Параметр «ip route» команды «Show»

Данная команда применяется для просмотра таблицы маршрутов.

Пример выполнения команды:

```
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
Gateway of last resort is 0.0.0.0 to network 0.0.0.0
C    192.168.1.0/24 is directly connected, FastEthernet0/0
C    192.168.2.0/24 is directly connected, Serial2/0
S    192.168.3.0/24 is directly connected, Serial2/0
S    192.168.4.0/24 is directly connected, Serial2/0
S    192.168.5.0/24 is directly connected, Serial2/0
S*   0.0.0.0/0 is directly connected, Serial2/0
Router#
```

Производится вывод таблицы маршрутизации.

Параметр «ip protocols» команды «Show»

Данная команда используется для просмотра протоколов маршрутизации, включенных на данном устройстве.

Пример выполнения команды:

```
Router#show ip protocols
Routing Protocol is "rip"
Sending updates every 30 seconds, next due in 18 seconds
Invalid after 180 seconds, hold down 180, flushed after 240
Outgoing update filter list for all interfaces is not set
Incoming update filter list for all interfaces is not set
Redistributing: rip
Default version control: send version 1, receive any version
  Interface          Send  Recv  Triggered RIP  Key-chain
  FastEthernet0/0    1     2 1
  Serial2/0          1     2 1
Automatic network summarization is in effect
Maximum path: 4
Routing for Networks:
        192.168.1.0
        192.168.2.0
Passive Interface(s):
Routing Information Sources:
        Gateway         Distance      Last Update
        192.168.2.2          120
Distance: (default is 120)
Router#
```

Выводится информация о включенных протоколах маршрутизации.

Команда «Ping»

Для проверки связи между устройствами сети можно использовать данную команду. Она отправляет эхо-запросы указанному узлу сети и фиксирует поступающие ответы.

Формат команды:

ping *A.B.C.D*

```
Router#ping 77.134.25.133
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 77.134.25.133, timeout is 2 seconds:
..!!!
Success rate is 60 percent (3/5)
```

Каждый ICMP-пакет, на который был получен ответ, обозначается восклицательным знаком, каждый потерянный пакет – точкой.

Контекст пользователя

Команда «Enable»

Выполнение конфигурационных или управляющих команд требует вхождения в привилегированный режим, используя данную команду.

Пример выполнения команды:

```
Router>enable
Router#
```

При вводе команды маршрутизатор перешел в привилегированный режим. Для выхода из данного режима используется команда disable или exit.

Также следует отметить, что в данном контексте можно пользоваться командой show для просмотра некоторой служебной информации.

Контекст конфигурирования маршрутизации

Команда «Network»

Данной командой указывают адреса сетей, которые будут доступны данному маршрутизатору.

Формат команды:

network *A.B.C.D* , где A.B.C.D – адрес сети

Пример выполнения команды:

```
Router(config-router)#network 192.168.3.0
```

Данная команда означает, что пакеты, направленные в подсеть 192.168.3.0 будут отправляться через данный шлюз.

2. Порядок выполнения работы.

Сеть, которую должен спроектировать студент, изображена на рис.1.5.

Как известно, локальная вычислительная сеть – это компьютерная сеть, покрывающая обычно относительно небольшую территорию или небольшую группу зданий. В нашем случае это всего-навсего 6 рабочих станций, определенным образом связанных между собой. Для этого мы будем использовать сетевые концентраторы (хабы) и коммутаторы (свичи).

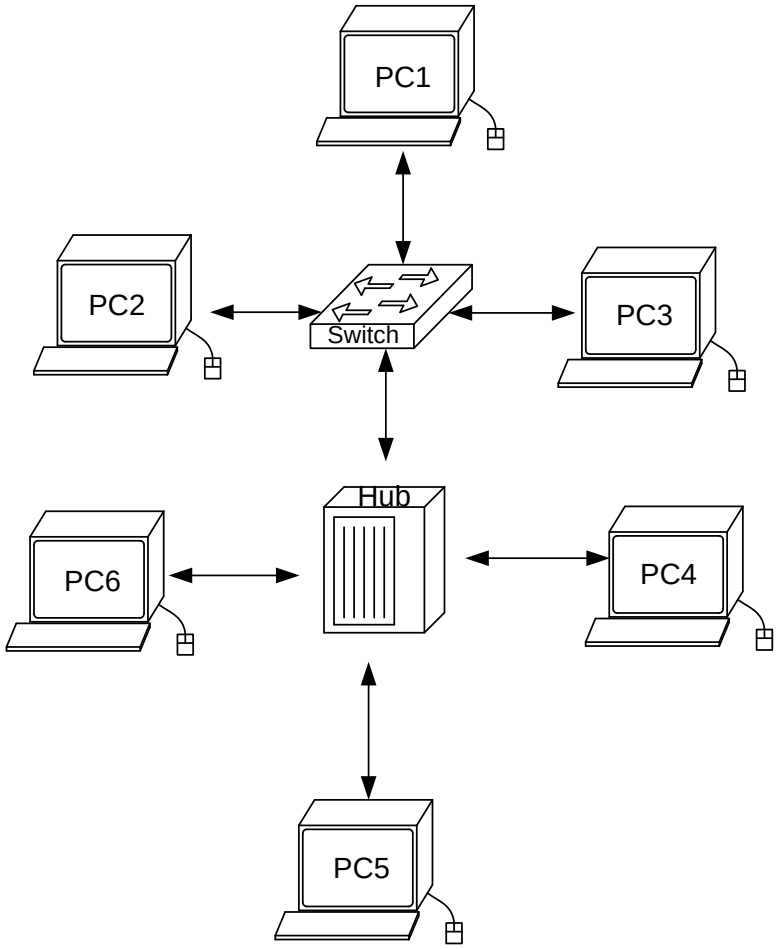


Рис.1.5. Проектируемая сеть

1.В нижнем левом углу Packet Tracer 4.0 выбираем устройства «Сетевые коммутаторы», и, в списке справа, выбираем коммутатор 2950-24,нажимая на него левой кнопкой мыши, вставляем его в рабочую область. Так же поступает с «Сетевым концентратором (Hub-PT)» и «Рабочими станциями (PC-PT)».

2.Далее необходимо соединить устройства, как показано на рис.1, используя соответствующий интерфейс. Для упрощения выбираем в нижнем левом углу Packet Tracer 4.0 «Тип связи» и указываем «Автоматически выбрать тип соединения»: нажимая на данный значок левой кнопкой мыши, затем нажимаем на необходимое нам устройство, и соединяем с другим все тем-же нажатием.

3.Далее идет самый важный этап – настройка. Так как мы используем устройства, работающие на начальных уровнях сетевой модели OSI (коммутатор на 2ом, концентратор – на 1ом), то их настраивать не надо. Необходима лишь настройка рабочих станций, а именно: IP-адреса, маски подсети, шлюза.

Ниже приведена настройка лишь одной станции (PC1) – остальные настраиваются аналогично. Производим двойной щелчок по нужной рабочей станции, в открывшемся окне выбираем вкладку Рабочий стол, далее – Конфигурация интерфейса, и производим соответствующую настройку:
IP-адрес. Как известно, в локальных сетях, основанных на протоколе IP, могут использоваться следующие адреса:

- 10.0.0.0—10.255.255.255;
- 172.16.0.0—172.31.255.255;
- 192.168.0.0—192.168.255.255.

Поэтому выбираем IP-адрес из данных диапазонах, например 192.168.0.1

Важно! IP-адреса всех рабочих станций должны находиться в одной и той-же подсети (то есть из одного диапазона), иначе процесс ping не выполнится.

Маска подсети. Значение подставится автоматически, когда будет введен IP-адрес.
Шлюз. Поле можно не заполнять.

4.Когда настройка завершена, можно переходить ко второй части работы – к запуску ping-процесса. Например, запускать его будем с PC5 и проверять наличие связи с PC1.

Важно! Студент сам может выбрать, откуда ему запускать ping-процесс, главное, чтобы выполнялось условие: пакеты должны обязательно пересылаться через коммутатор и концентратор.

Для этого производим двойной щелчок по нужной рабочей станции, в открывшемся окне выбираем вкладку «Рабочий стол», далее – «Командная строка». Нам предлагают ввести команду, что мы и делаем:

```
PC>ping 192.168.0.1
```

и жмем клавишу Enter. Если все настроено верно, то мы увидим следующую информацию:

```
Pinging 192.168.0.1with 32 bytes of data:
Reply from 192.168.0.1: bytes=32 time=183ms TTL=120
Reply from 192.168.0.1: bytes=32 time=90ms TTL=120
Reply from 192.168.0.1: bytes=32 time=118ms TTL=120
Reply from 192.168.0.1: bytes=32 time=87ms TTL=120
Ping statistics for 192.168.0.1:
   Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
   Minimum = 87ms, Maximum = 183ms, Average = 119ms
PC>
```

Это означает, что связь установлена, и данный участок сети работает исправно.

5. В Packet Tracer 4.0 предусмотрен режим моделирования, в котором подробно описывается и показывается, как работает утилита Ping. Поэтому необходимо перейти в данный режим, нажав на одноименный значок в нижнем левом углу рабочей области, или по комбинации клавиш Shift+S. Откроется «Панель моделирования», в которой будут отображаться все события, связанные с выполнения ping-процесса.

Теперь необходимо повторить запуск ping-процесса. После его запуска можно сдвинуть «Панель моделирования», чтобы на схеме спроектированной сети наблюдать за отправкой/приемкой пакетов.

Кнопка «Автоматически» подразумевает моделирование всего ping-процесса в едином процессе, тогда как «Пошагово» позволяет отображать его пошагово.

Чтобы узнать информацию, которую несет в себе пакет, его структуру, достаточно нажать правой кнопкой мыши на цветной квадрат в графе «Информация».

Моделирование прекращается либо при завершении ping-процесса, либо при закрытии окна «Редактирования» соответствующей рабочей станции.

3. Содержание отчета

1) титульный лист;
2) формулировку цели работы;
3) описание результатов выполнения;
4) выводы, согласованные с целью работы.

ЛАБОРАТОРНАЯ РАБОТА №2

Понятие сетевых протоколов. Стек протоколов TCP/IP.

Цель:

- Изучить понятие и назначение сетевых протоколов.
- Изучить эталонную модель взаимодействия OSI/ISO.
- Изучить стек протоколов TCP/IP и его сопоставление с моделью OSI.
- Изучить протоколы сетевого и транспортного уровней: IP, TCP, UDP.
- Выяснить назначение документации RFC.
- Знать принципы реализации протоколов на базе машины состояний.

1. Общие сведения.

Сетевой протокол в компьютерных сетях – набор правил для специфического типа связи.

Разные протоколы зачастую описывают лишь разные стороны одного типа связи; взятые вместе, они образуют стек протоколов. Названия «протокол» и «стек протоколов» также указывают на программное обеспечение, которым реализуется протокол.

Новые протоколы для Интернета определяются IETF (Internet Engineering Task Force – проблемная группа проектирования Internet), а прочие протоколы – IEEE или ISO. ITU-T (International Telecommunication Union, ITU) занимается телекоммуникационными протоколами и форматами.

Эталонная модель OSI

Чтобы помочь поставщикам в стандартизации и интеграции их сетевого программного обеспечения, международная организация по стандартизации (ISO) определила программную модель пересылки сообщений между компьютерами. Эта модель получила название *эталонной модели OSI* (Open Systems Interconnection). В ней определено семь уровней программного обеспечения (рис. 2.1).

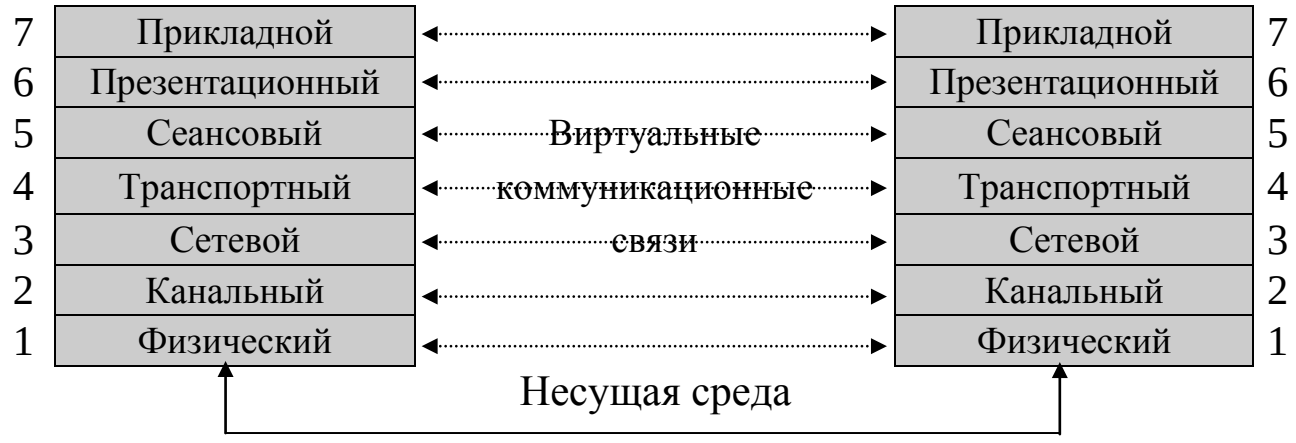


Рис. 2.1. Эталонная модель OSI

Эталонная модель OSI – идеал, точно реализованный лишь в очень немногих системах, но часто используемый при объяснении основных принципов работы сети. Каждый уровень на одной из машин считает, что он взаимодействует с тем же уровнем на другой машине. На данном уровне обе машины «разговаривают» на одном языке, или протоколе. Но в действительности сетевой запрос должен сначала пройти до самого нижнего уровня на первой машине, затем он передается по несущей среде и уже на второй машине вновь поднимается до уровня, который его поймет и обработает.

Задача каждого уровня в том, чтобы предоставить сервисы более высоким уровням и скрывать от них конкретную реализацию этих сервисов.

Прикладной уровень (Application layer). Верхний (7-й) уровень модели, обеспечивает взаимодействие сети и пользователя. Уровень разрешает доступ к сетевым службам приложениям пользователя, таким как обработчик запросов к базам данных, доступ к файлам, пересылке электронной почты. Также отвечает за передачу служебной информации, предоставляет приложениям информацию об ошибках и формирует запросы к уровню представления.

Уровень представления (Presentation layer). Этот уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Запросы приложений, полученные с уровня приложений, он преобразует в формат для передачи по сети, а полученные из сети данные преобразует в формат, понятный приложениям. На этом уровне может осуществляться сжатие/распаковка или кодирование/раскодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

Сеансовый уровень (Session layer). Отвечает за поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизации задач, определением права на передачу данных и поддержание сеанса в периоды неактивности приложений. Синхронизация передачи обеспечивается помещением в поток данных контрольных точек, начиная с которых возобновляется процесс при нарушении взаимодействия.

Транспортный уровень (Transport layer). 4-й уровень модели, предназначен для доставки данных без ошибок, потерь и дублирования в той последовательности, как они были переданы. При этом неважно какие данные передаются, откуда и куда, то есть он предоставляет сам механизм передачи. Блоки данных он разделяет на фрагменты, размер которых зависит от протокола, короткие объединяет в один, длинные разбивает. Протоколы этого уровня предназначены для взаимодействия типа точка-точка.

Сетевой уровень (Network layer). 3-й уровень сетевой модели OSI, предназначен для определения пути передачи данных. Отвечает за трансляцию логических адресов и имён в физические, определение кратчайших маршрутов, коммутация и маршрутизация пакетов, отслеживание неполадок и заторов в сети. На этом уровне работает такое сетевое устройство, как маршрутизатор.

Канальный уровень (Data Link layer). Этот уровень предназначен для обеспечения взаимодействия сетей на физическом уровне и контроле за ошибками, которые могут возникнуть. Полученные данные от физического уровня он упаковывает в кадры данных, проверяет на целостность, если нужно исправляет ошибки и отправляет на сетевой уровень. Канальный уровень может взаимодействовать с одним или несколькими физическими уровнями, контролируя и управляя этим взаимодействием. Спецификация IEEE 802 разделяет этот уровень на 2 подуровня – MAC (Media Access Control) регулирует доступ к разделяемой физической среде, и LLC (Logical Link Control) обеспечивает обслуживание сетевого уровня. На этом уровне работают коммутаторы, мосты и сетевые адаптеры.

В программировании этот уровень представляет драйвер сетевой платы, в операционных системах имеется программный интерфейс взаимодействия канального и сетевого уровня между собой, это не новый уровень, а просто реализация модели для конкретной ОС. Примеры таких интерфейсов: ODI, NDIS.

Физический уровень (Physical layer). Самый нижний уровень модели, предназначен непосредственно для передачи потока данных. Осуществляет передачу электрических или

оптических сигналов в кабель и соответственно их приём и преобразование в биты данных в соответствии с методами кодирования цифровых сигналов. Другими словами осуществляет интерфейс между сетевым носителем и сетевым устройством. На этом уровне работают концентраторы и повторители (ретрансляторы) сигнала.

Инкапсуляция и обработка пакетов

При продвижении пакета данных по уровням сверху вниз каждый новый уровень добавляет к пакету свою служебную информацию в виде заголовка и, возможно, трейлера (информации, помещаемой в конец сообщения). Эта операция называется инкапсуляцией данных верхнего уровня в пакете нижнего уровня. Служебная информация предназначается для объекта того же уровня на удаленном компьютере, ее формат и интерпретация определяются протоколом данного уровня.

Разумеется, данные, приходящие с верхнего уровня, могут на самом деле представлять собой пакеты с уже инкапсулированными данными еще более верхнего уровня.

С другой стороны, при получении пакета от нижнего уровня он разделяется на заголовок (трейлер) и данные. Служебная информация из заголовка (трейлера) анализируется и в соответствии с ней данные, возможно, направляются одному из объектов верхнего уровня. Тот в свою очередь рассматривает эти данные как пакет со своей служебной информацией и данными для еще более верхнего уровня, и процедура повторяется, пока пользовательские данные, очищенные от всей служебной информации, не достигнут прикладного процесса.

Возможно, что пакет данных не будет доведен до самого верхнего уровня, например, в случае, если данный компьютер представляет собой промежуточную станцию на пути между отправителем и получателем. В этом случае объект соответствующего уровня при анализе служебной информации заметит, что пакет на этом уровне адресован не ему (хотя с точки зрения нижележащих уровней он был адресован именно этому компьютеру). Тогда объект выполнит необходимые действия для перенаправления пакета к месту назначения или возврата отправителю с сообщением об ошибке, но в любом случае не будет продвигать данные на верхний уровень.

Применимость модели OSI

Хотя модель OSI полезна как основа для обсуждения сетевых архитектур и реализаций, ее нельзя рассматривать как готовый чертеж для создания любой сетевой архитектуры. Не следует также думать, что размещение некоторой функции на уровне N в этой модели означает, что только здесь наилучшее для нее место.

Модель OSI имеет множество недостатков. Хотя, в конечном итоге, были созданы работающие реализации, протоколы OSI на сегодняшний день утратили актуальность. Основные проблемы этой модели в том, что, во-первых, распределение функций между уровнями произвольно и не всегда очевидно, во-вторых, она была спроектирована (комитетом) без готовой реализации.

Другая проблема модели OSI – это сложность и неэффективность. Некоторые функции выполняются сразу на нескольких уровнях. Так, обнаружение и исправление ошибок происходит на большинстве уровней.

Наконец, выбор именно семи уровней продиктован, скорее, политическими, а не техническими причинами. В действительности, сеансовый уровень и уровень представления редко встречаются в реально работающих сетях.

Спецификации протоколов. Документация RFC

Спецификации практических реализаций сетевых протоколов и связанные с ними архитектурные вопросы (в частности, Internet) содержатся в серии документов, объединенных названием Request for Comments (RFC – Предложения для обсуждения). На самом деле, RFC, впервые появившиеся в 1969 году, – это не только спецификации протоколов. Их можно назвать рабочими документами, в которых обсуждаются разнообразные аспекты компьютерных

коммуникаций и сетей. Не все RFC чисто технические, в них встречаются забавные наблюдения, пародии стихи и просто различные высказывания. К концу 1999 года было более 2000 присвоенных RFC номеров, правда, некоторые из них так и не были опубликованы.

Хотя не в каждом RFC содержится какой-либо стандарт Internet, любой стандарт Internet опубликован в виде RFC. Материалам, входящим в подсерию RFC дается дополнительная метка «STDxxxx». Текущий список стандартов и тех RFC которые находятся на пути принятия в качестве стандарта, опубликован в документе STD0001.

Не следует, однако, думать, что RFC, не упомянутые в документе STD0001 лишены технической ценности. В некоторых описываются идеи пока еще разрабатываемых протоколов или направления исследовательских работ. Другие содержат информацию или отчеты о деятельности многочисленных рабочих групп, созданных по решению IETF (Internet Engineering Task Force - проблемная группа проектирования Internet).

Получить копии RFC можно разными путями, но самый простой - зайти на Web-страницу редактора RFC http://www.rfc-editor.org. На этой странице есть основанное на заполнении форм средство загрузки, значительно упрощающее поиск. Есть также поиск по ключевым словам, позволяющий найти нужные RFC, если их номер неизвестен. Там же можно получить документы из подсерий STD, FYI и BCP (Best Current Practices - лучшие современные решения).

После публикации ни номер, ни текст RFC уже не изменяются, так что единственный способ модифицировать RFC – это выпустить другое RFC, заменяющее предыдущее. Для каждого RFC в указателе отмечено, есть ли для него заменяющее RFC и если есть, то его номер. Там же указаны RFC, которые обновляют, но не замещают прежние.

Стек протоколов TCP/IP

TCP/IP – собирательное название для набора (стека) сетевых протоколов разных уровней, используемых в Интернет. Особенности TCP/IP:

- открытые стандарты протоколов, разрабатываемые независимо от программного и аппаратного обеспечения;
- независимость от физической среды передачи;
- система уникальной адресации;
- стандартизованные протоколы высокого уровня для распространенных пользовательских сервисов.

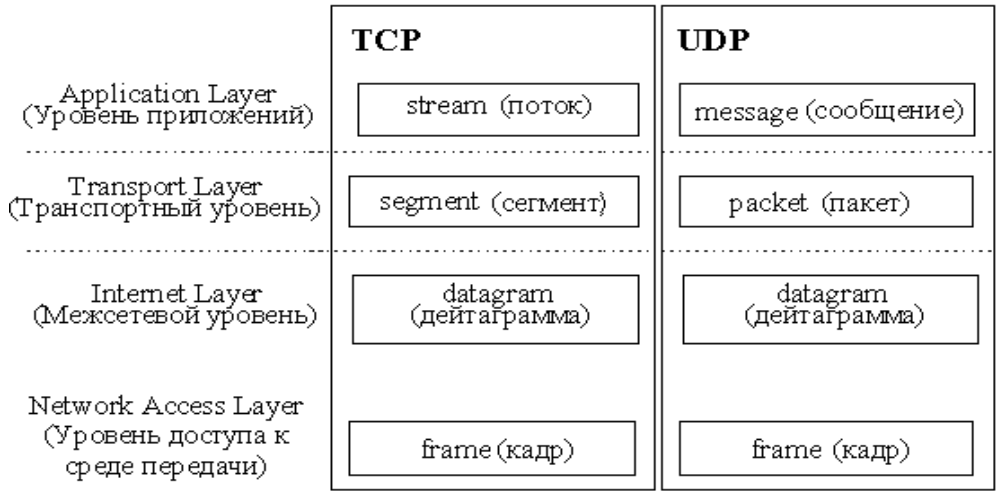


Рис. 2.2. Уровни стека протоколов TCP/IP

Стек протоколов TCP/IP делится на 4 уровня: прикладной (application), транспортный

(transport), межсетевой (internet) и уровень доступа к среде передачи (network access). Термины, применяемые для обозначения блока передаваемых данных, различны при использовании разных протоколов транспортного уровня – TCP и UDP, поэтому на рисунке 2 изображено два стека. Как и в модели OSI, данные более верхних уровней инкапсулируются в пакеты нижних уровней (см. рис. 2.3).

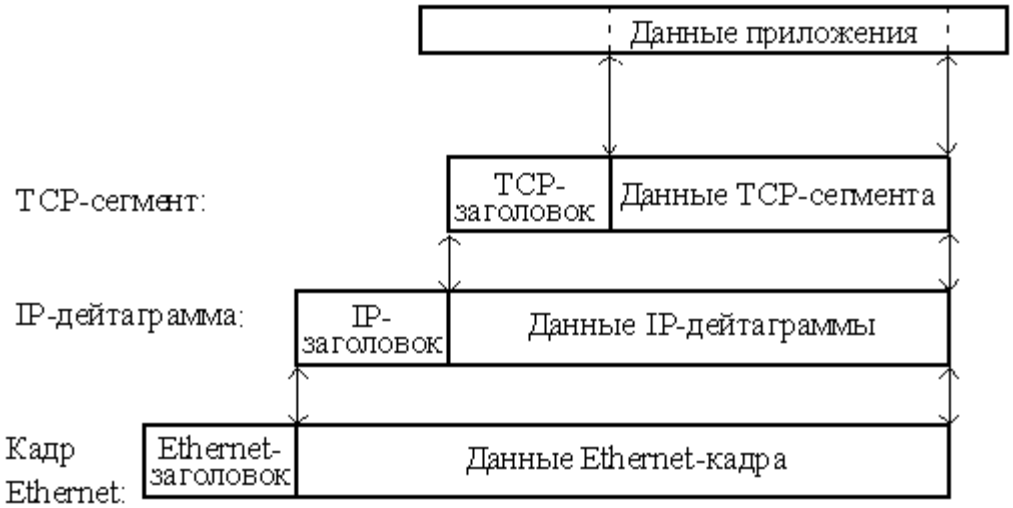


Рис. 2.3. Пример инкапсуляции пакетов в стеке TCP/IP

Примечание. Принцип функционирования протоколов в стеке TCP/IP (собственно говоря, это справедливо и для остальных протоколов) никак не зависит от операционной системы!

Рис. 2.4. Соотношение уровней стеков OSI и TCP/IP

Ниже кратко рассматриваются функции каждого уровня и примеры протоколов. Программа, реализующая функции того или иного протокола, часто называется модулем, например, «IP-модуль», «модуль TCP».

Уровень приложений. Приложения, работающие со стеком TCP/IP, могут также выполнять функции уровней представления и частично сеансового модели OSI; например, преобразование данных к внешнему представлению, группировка данных для передачи и т.п.

Распространенными примерами приложений являются программы telnet, ftp, HTTP-серверы и клиенты, программы работы с электронной почтой и др.

Для пересылки данных другому приложению, приложение обращается к тому или иному модулю транспортного уровня.

Транспортный уровень. Протоколы транспортного уровня обеспечивают прозрачную (сквозную) доставку данных (end-to-end delivery service) между двумя прикладными процессами. Процесс, получающий или отправляющий данные с помощью транспортного уровня, идентифицируется на этом уровне номером, который называется номером порта. Таким образом, роль адреса отправителя и получателя на транспортном уровне выполняет номер *порта* (см. далее).

Анализируя заголовок своего пакета, полученного от межсетевого уровня, транспортный модуль определяет по номеру порта получателя, какому из прикладных процессов направлены данные, и передает эти данные соответствующему прикладному процессу (возможно, после проверки их на наличие ошибок и т.п.). Номера портов получателя и отправителя записываются в заголовок транспортным модулем, отправляющим данные; заголовок транспортного уровня содержит также и другую служебную информацию; формат заголовка зависит от используемого транспортного протокола.

На транспортном уровне работают два основных протокола: UDP и TCP.

TCP (Transmission Control Protocol – протокол контроля передачи, *RFC 793*) – это транспортный механизм, предоставляющий поток данных, с предварительной установкой соединения, за счёт этого дающий уверенность в безошибочности получаемых данных, осуществляет повторный запрос данных в случае потери пакетов и устраняет дублирование при получении двух копий одного пакета. Естественно, что в общем случае данные не могут быть гарантировано доставлены до адресата; в таком случае клиентский процесс получает об этом уведомление.

Данными для TCP является не интерпретируемая протоколом последовательность пользовательских октетов, разбиваемая для передачи по частям. Каждая часть передается в отдельном TCP-сегменте. Для продвижения сегмента по сети между компьютером-отправителем и компьютером-получателем модуль TCP пользуется сервисом межсетевого уровня (вызывает модуль IP). Протокол TCP гарантирует, что приложение получит данные точно в такой же последовательности, в какой они были отправлены, и без потерь.

Более подробно работу протокола TCP будем рассматривать на последующих занятиях.

UDP (User Datagram Protocol, протокол пользовательских дейтаграмм, *RFC* 768) фактически не выполняет каких-либо особых функций дополнительно к функциям межсетевого уровня (протокола IP см. далее). Протокол UDP используется либо при пересылке коротких сообщений, когда накладные расходы на установление сеанса и проверку успешной доставки данных оказываются выше расходов на повторную (в случае неудачи) пересылку сообщения, либо в том случае, когда сама организация процесса-приложения обеспечивает установление соединения и проверку доставки пакетов.

Пользовательские данные, поступившие от прикладного уровня, предваряются UDP-заголовком, и сформированный таким образом UDP-пакет отправляется на межсетевой уровень.

Межсетевой уровень и протокол IP. Основным протоколом этого уровня является протокол IP (Internet Protocol, *RFC 791*).

Протокол IP доставляет блоки данных, называемых дейтаграммами, от одного сетевого узла к другому.

В современной сети Интернет используется IP четвёртой версии, также известный как IPv4. В протоколе IP этой версии каждому узлу сети ставится в соответствие IP-адрес длиной 4 октета (иногда говорят «байта», подразумевая распространённый восьмибитовый минимальный адресуемый фрагмент памяти ЭВМ). Более подробно об IP-адресах протокола 4-й версии можно прочитать в предыдущей лабораторной работе.

В настоящее время вводится в эксплуатацию шестая версия протокола — IPv6, которая позволяет адресовать значительно большее количество узлов, чем IPv4. Эта версия отличается повышенной разрядностью адреса, встроенной возможностью шифрования и некоторыми другими особенностями. Переход с IPv4 на IPv6 связан с трудоёмкой работой операторов связи и производителей программного обеспечения и не может быть выполнен одномоментно. На начало 2007 года в Интернете присутствовало около 760 сетей, работающих по протоколу IPv6. Для

сравнения, на то же время в адресном пространстве IPv4 присутствовало более 203 тысяч сетей, но в IPv6 сети гораздо более крупные, нежели в IPv4.

Данные для IP дейтаграммы передаются IP-модулю транспортным уровнем. IP-модуль предваряет эти данные заголовком, содержащим IP-адреса отправителя и получателя и другую служебную информацию, и сформированная таким образом дейтаграмма передается на уровень доступа к среде передачи (например, одному из физических интерфейсов) для отправки по каналу передачи данных.

Не все сетевые узлы могут непосредственно связаться друг с другом; часто для того, чтобы передать дейтаграмму по назначению, требуется направить ее через один или несколько промежуточных узлов по тому или иному маршруту. Задача определения маршрута для каждой дейтаграммы решается протоколом IP.

Когда модуль IP получает дейтаграмму с нижнего уровня, он проверяет IP-адрес назначения. Если дейтаграмма адресована данному компьютеру, то данные из нее передаются на обработку модулю вышестоящего уровня (какому конкретно – указано в заголовке дейтаграммы). Если же адрес назначения дейтаграммы – чужой, то модуль IP может принять два решения: первое – уничтожить дейтаграмму, второе – отправить ее дальше к месту назначения, определив маршрут следования – так поступают промежуточные станции – маршрутизаторы.

Также может потребоваться, на границе сетей с различными характеристиками, разбить дейтаграмму на фрагменты, а потом собрать в единое целое на компьютере-получателе. Это тоже задача протокола IP.

Если модуль IP по какой-либо причине не может доставить дейтаграмму, она уничтожается. При этом модуль IP может отправить компьютеру-источнику этой дейтаграммы уведомление об ошибке; такие уведомления отправляются с помощью протокола ICMP, являющегося неотъемлемой частью модуля IP. Более никаких средств контроля корректности данных, подтверждения их доставки, обеспечения правильного порядка следования дейтаграмм, предварительного установления соединения между компьютерами протокол IP не имеет. Эта задача возложена на транспортный уровень.

Более подробную информацию об IP-адресах можно найти в предыдущей лабораторной работе.

Уровень доступа к среде передачи. Функции этого уровня:

- отображение IP-адресов в физические адреса сети (MAC-адреса, например, Ethernet-адрес в случае сети Ethernet). Эту функцию выполняет протокол ARP;
- инкапсуляция IP-дейтаграмм в кадры для передачи по физическому каналу и извлечение дейтаграмм из кадров. При этом не требуется какого-либо контроля безошибочности передачи (хотя он может и присутствовать), поскольку в стеке TCP/IP такой контроль возложен на транспортный уровень или на само приложение. В заголовке кадров указывается точка доступа к сервису (SAP, Service Access Point) - поле, содержащее код протокола межсетевого уровня, которому следует передать содержимое кадра (в нашем случае это протокол IP);
- определение метода доступа к среде передачи - то есть способа, с помощью которого компьютер устанавливает свое право на произведение передачи данных (передача токена, опрос компьютеров, множественный доступ с детектированием коллизий и т.п.).
- определение представления данных в физической среде;
- пересылка и прием кадра.

Стек TCP/IP не подразумевает использования каких-либо определенных протоколов уровня доступа к среде передачи и физических сред передачи данных. От уровня доступа к среде передачи требуется наличие интерфейса с модулем IP, обеспечивающего передачу дейтаграммы между уровнями. Также требуется обеспечить преобразование IP-адреса узла сети, на который передается дейтаграмма, в MAC-адрес. Часто в качестве уровня доступа к среде передачи могут выступать целые протокольные стеки, тогда говорят об IP поверх ATM, IP поверх IPX, IP поверх X.25 и т.п.

Обобщенная модель взаимодействия узлов на базе протоколов TCP/IP представлена на рис 2.5.

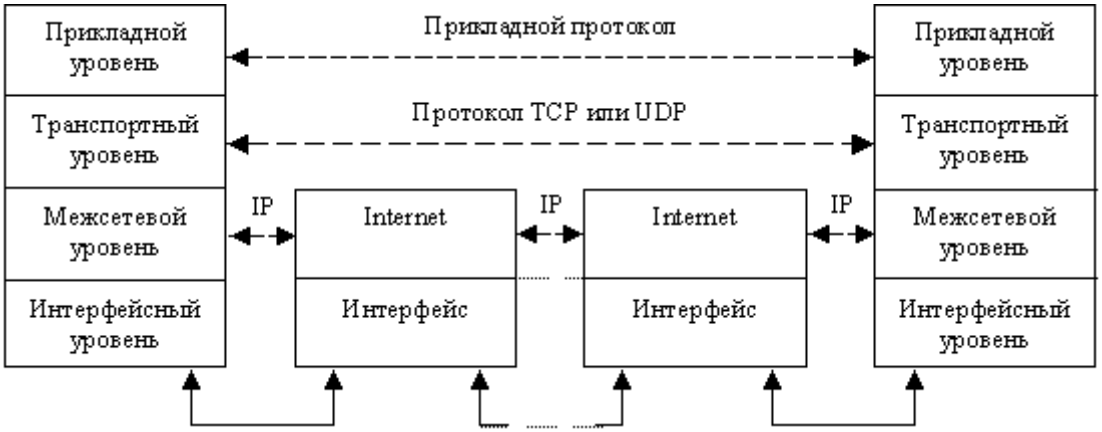


Рис. 2.5. Модель взаимодействия стеков TCP/IP

Понятие сетевых портов и сокетов

Как уже было сказано, основные прикладные сетевые сервисы используют средства транспортного уровня для взаимодействия.

Любые 2 сетевых процесса могут идентифицировать друг друга при помощи 3-х компонент: ip-адрес, протокол(TCP/UDP), порт. Часто данные компоненты носят название ***сокетами***. Сокеты – это название программного интерфейса для обеспечения информационного обмена между процессами. Т.е. для прикладных сетевых процессов взаимодействие осуществляется через сокеты. Более детально сокеты мы будем изучать на последующих занятиях. Сейчас же рассмотрим понятие портов более подробно.

Порт – параметр протоколов TCP и UDP, определяющий пункт назначения для данных, принимаемых по сети. Порту сопоставляется номер от 1 до 65535, позволяющие различным программам, выполняемым на одном хосте, получать данные независимо друг от друга. В этом случае каждая из них обрабатывает данные, поступающие на определённый порт (иногда говорят, что программа «слушает» на том или ином порту).

Согласно IP, в каждом пакете присутствуют IP адрес узла-источника и IP афдрес узла-назначения. В TCP/UDP пакетах дополнительно указываются порт источника и порт назначения. Узел назначения, получив пакет, смотрит на порт назначения и передает пакет соответствующему у себя приложению. Использование портов позволяет независимо использовать TCP/UDP протокол сразу многим приложениям на одном и том же компьютере.

Для сетевых приложений нотация указания порта следующая: «ip**:**port». Например, http://web-service.org:8888

Пояснение понятия портов представлено на рис.2. 6. На самом деле сетевой порт – это

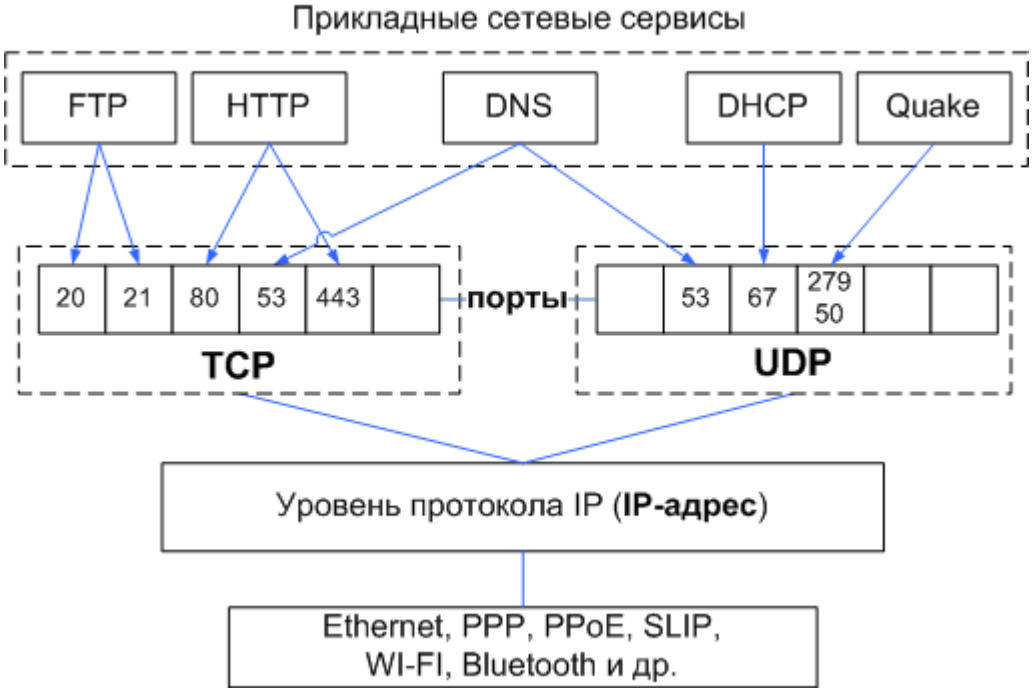


Рис. 2.6. Компоненты сокетов

всего лишь числовой параметр в сетевом пакете протоколов TCP и UDP. Такие понятия как «открыть порт» означают что пакеты адресованные на данный порт будут приниматься на обработку.

Порты из диапазона 1-1024 являются привилегированными. Называются они так, потому что для их открытия (и, соответственно, запуска соответствующих сетевых сервисов) на большинстве ОС требуются права системного администратора. Большая часть привилегированных портов распределена для общеупотребительных сетевых протоколов. В табл. 1 перечислены

некоторые протоколы и порты, за которыми они закреплены. Данные порты являются портами по умолчанию для соответствующих служб и чаще всего не перенастраиваются.

Таблица 2.1

Примеры некоторых стандартных сетевых портов

Порт / Протокол	Сервис	Описание
20/TCP	ftp-data	Порт данных FTP
21/TCP	ftp	Порт протокола передачи файлов (File Transfer Protocol, FTP); иногда используется протоколом файловой службы (File Service Protocol, FSP)
22/TCP	ssh	Служба Безопасной ОБолочки (Secure SHell, SSH)
23/TCP	telnet	Служба Telnet
25/TCP	smtp	Протокол простой передачи почты (Simple Mail Transfer Protocol, SMTP)
53/(UDP,TCP)	domain	Службы доменных имён (такие как BIND)
80/TCP	http	Протокол передачи гипертекста (HyperText Transfer Protocol, HTTP) для служб всемирной паутины (World Wide Web, WWW)
110/TCP	pop3	Протокол почтового отделения (Post Office Protocol) версии 3
443/TCP	https	Протокол HTTP поверх SSL
992/TCP	telnets	Telnet поверх SSL (TelnetS)
993/TCP	imaps	IMAP поверх SSL (IMAPS)
994/TCP	ircs	IRC поверх SSL (IRCS)
995/TCP	pop3s	POP 3 поверх SSL (POP3S)

История стека протоколов TCP/IP

История Internet и его протоколов началась в 1961 году, когда Леонард Клайнрок разработал в MIT теорию коммутации пакетов. Его работа была основана на идее о разделении данных между множеством небольших пакетов и отправке их на место назначения отдельно, без указания точного пути. После первоначального скептического отношения этот принцип был в конечном счете использован в исследовательском проекте ARPA (Advanced Research Projects Agency), отделения Министерства обороны США. В 1968 году ARPA выделила бюджет более чем в полмиллиона долларов для гетерогенных сетей, которые были названы ARPANET.

В 1969 году эта экспериментальная сеть соединяла 4 университета: Лос-Анжелеса (UCLA), Санта-Барбары (UCSB), Юты и Стенфордский исследовательский институт (SRI) – и расширялась очень быстро. Позже к ARPANET были успешно подключены спутниковые и сотовые каналы связи.

Эта система интенсивно использовалась в последующие годы. На основе знаний, полученных из этой системы, было разработано второе поколение протоколов. К 1982 году был определен набор протоколов с двумя важными протоколами: TCP и IP. Сегодня название TCP/IP используется для всего набора протоколов. В 1983 году TCP/IP стал стандартным протоколом для ARPANET. Протоколы TCP/IP оказались особенно подходящими для обеспечения надежного соединения в сетях внутри постоянно растущего ARPANET. ARPA было очень заинтересовано в установлении новых протоколов и убедило Калифорнийский университет в Беркли интегрировать протоколы TCP/IP в их широко используемую операционную систему Berkley BSD UNIX. Для

проектирования приложений с сетевыми возможностями был использован принцип сокетов. Это помогло протоколам TCP/IP стать вскоре очень популярными для обмена между приложениями.

В последующие годы ARPENET выросла до таких размеров, которые сделали управление IP-адресами всех компьютеров в одном простом файле слишком дорогим. Как следствие, была разработана служба доменных имен (Domain Name Service – DNS), которая используется для сокрытия IP-адресов за легко запоминаемыми именами компьютеров и доменов. Сегодня чаще всего использующимся протоколом сетевого уровня является протокол Internet версии 4. Однако он не был спроектирован для такого огромного распространения и уже достиг пределов своих возможностей, поэтому пришлось разработать новую версию. Новый протокол Internet версии 6 называется также IPv6 или IPng.

Мониторинг сетевой активности и анализ работы сетевых приложений

Для закрепления знаний на практике рассмотрим с вами ряд полезных сетевых утилит, которые позволяют вести мониторинг состояния стека TCP/IP.

Первой утилитой, которую мы с вами рассмотрим будет утилита netstat. Данная утилита позволяет получать информацию об активных сетевых соединениях на уровне TCP и UDP протоколов, получать базовую статистику по количеству переданных и полученных пакетов уровня IP и т.д. Синтаксис команды можно почерпнуть из встроенной справки в саму утилиту («netstat /?»)

Рассмотрим несколько примеров.

«netstat -a» – получение информации обо всех установленных соединениях и открытых на прослушивание портах (см. рис. 2.7)

«netstat -n -b» – получение информации обо всех активных соединениях и процессах инициировавших их (см. рис. 2.8)

«netstat -e -s» – получение основной статистики по всем протоколам (ethernet, IPv4, IPv6, TCP, UDP) (см. рис. 2.9)

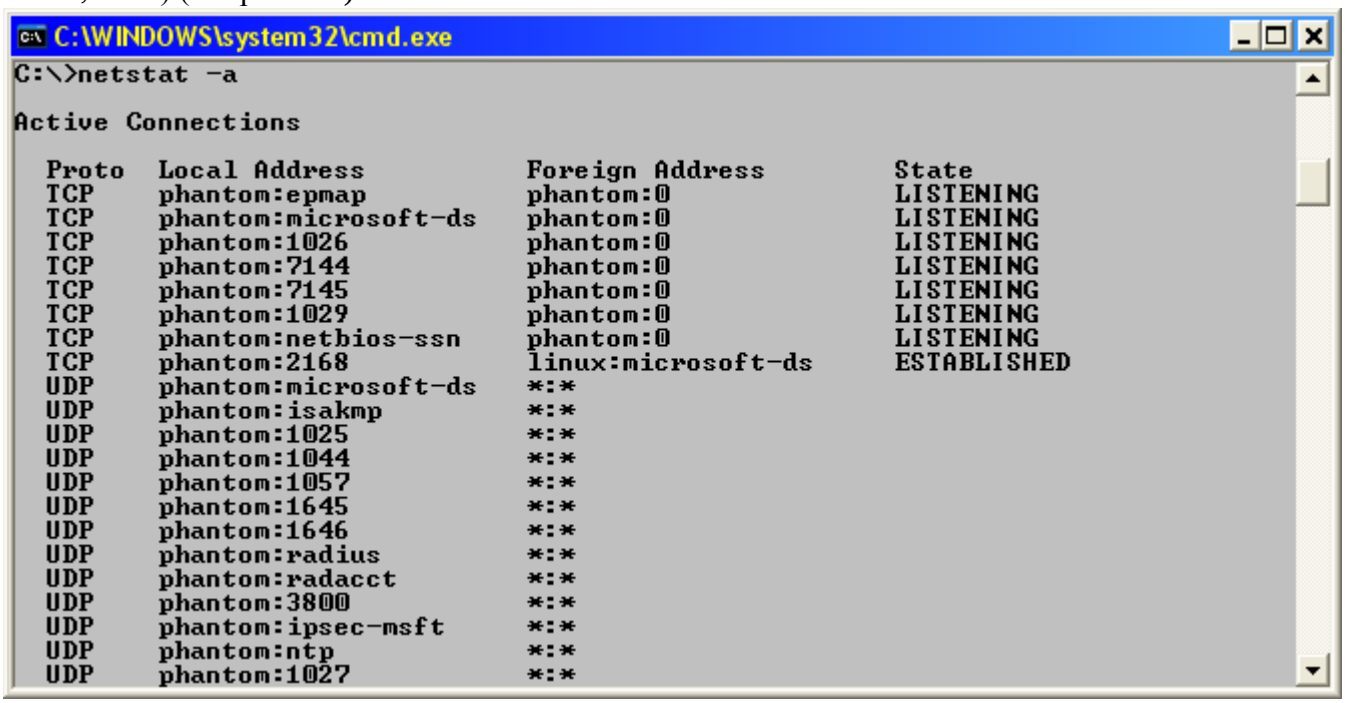


Рис. 2.7. Список всех установленных TCP/UDP соединений и открытых на прослушивание портов.

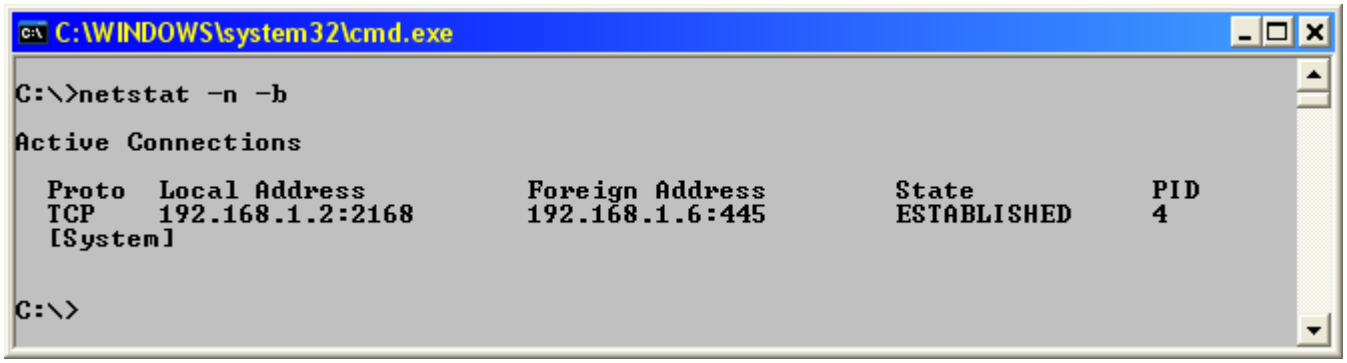


Рис. 2.8.. Список активных соединений с указанием процесса

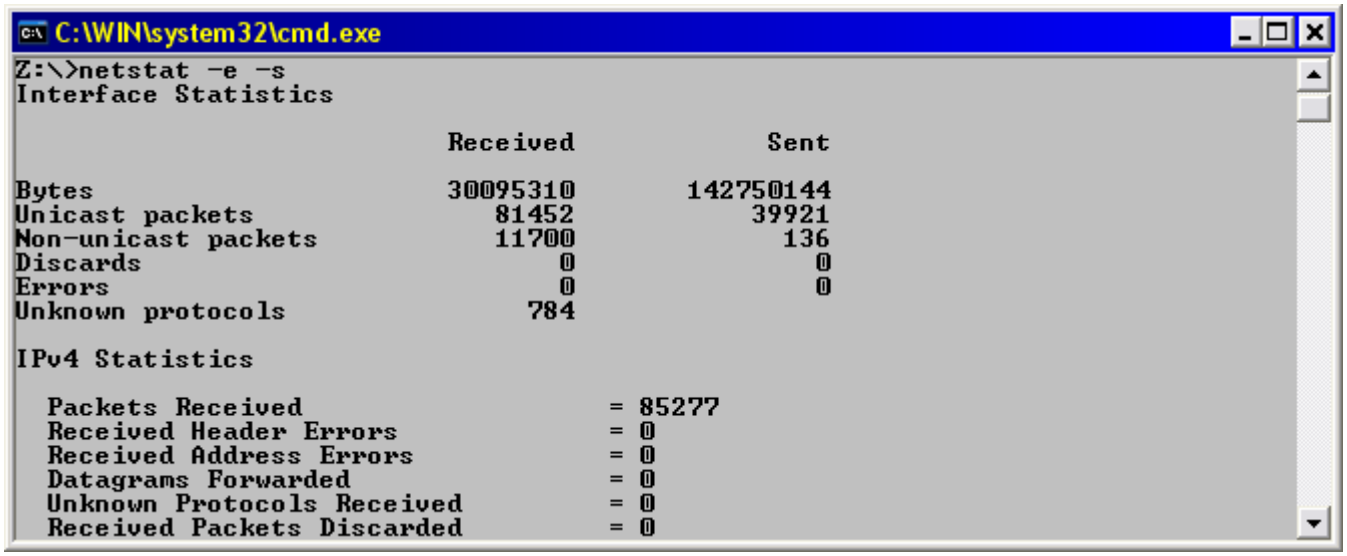


Рис.2. 9. Статистика по протоколам

Очень удобным аналогом утилиты netstat для отслеживания активных соединений является утилита TCPView (см.рис. 2.10). Она позволяет в интерактивном режиме отслеживать сетевые соединения, а также позволяет получать информацию о процессах, установивших соединение и завершать их в случае необходимости. Утилита является бесплатной и ее можно можно скачать с сайта фирмы Майкрософт (www.microsoft.com/technet/sysinternals/utilities/tcpview.mspx)

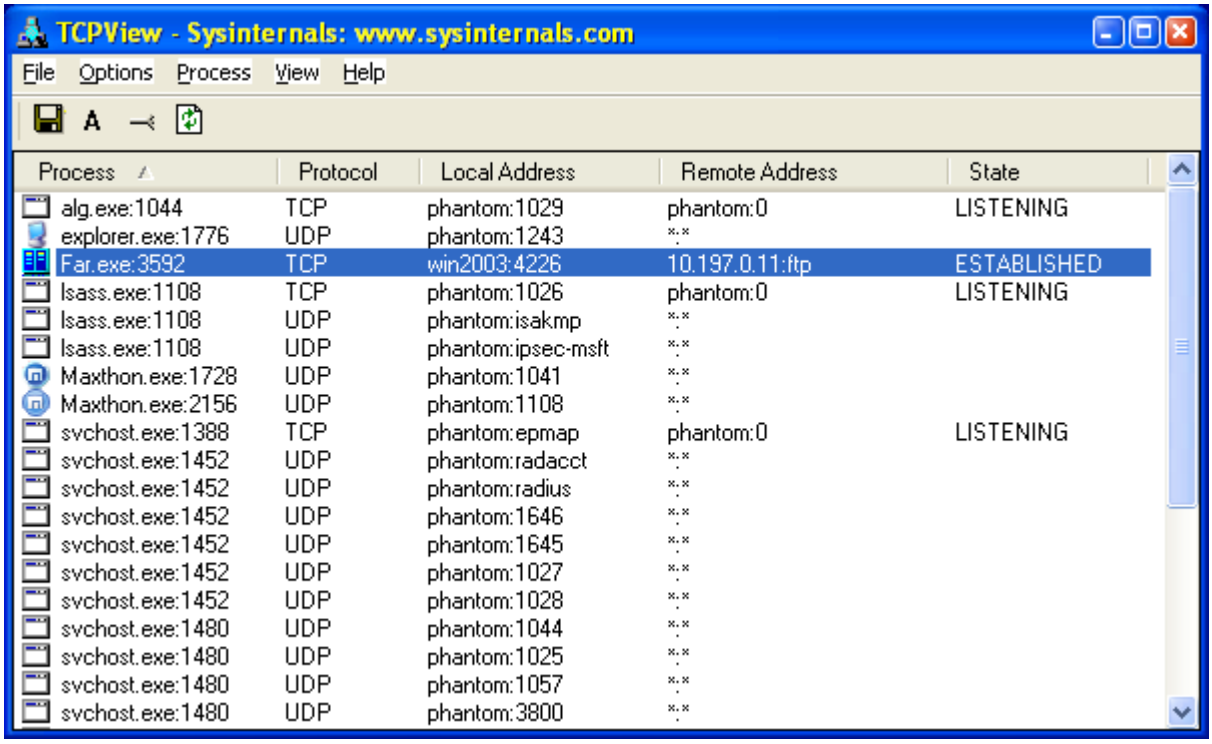


Рис. 2.10. Утилита TCPView

Сканеры портов

Часто возникает необходимость узнать, какие сетевые сервисы запущены на удаленной машине. Для решения данной задачи служат т.н. сканеры портов. Данная группа утилит позволяет с некоторой точностью узнать, какие порты открыты на удаленной машине и некоторые другие параметры. Зная номера портов зачатую можно с достаточно большой уверенностью предположить, какие сервисы запущены на удаленной машине.

Наиболее достоверная информация может быть получена для протокола TCP. Задача детектирования UDP-сервисов не всегда может быть решена. Поэтому высокая точность сканирования UDP-сервисов не гарантируется.

Одним из самых распространенных профессиональных сканеров портов является сканер «NMAP» (см. рис. 2.11)

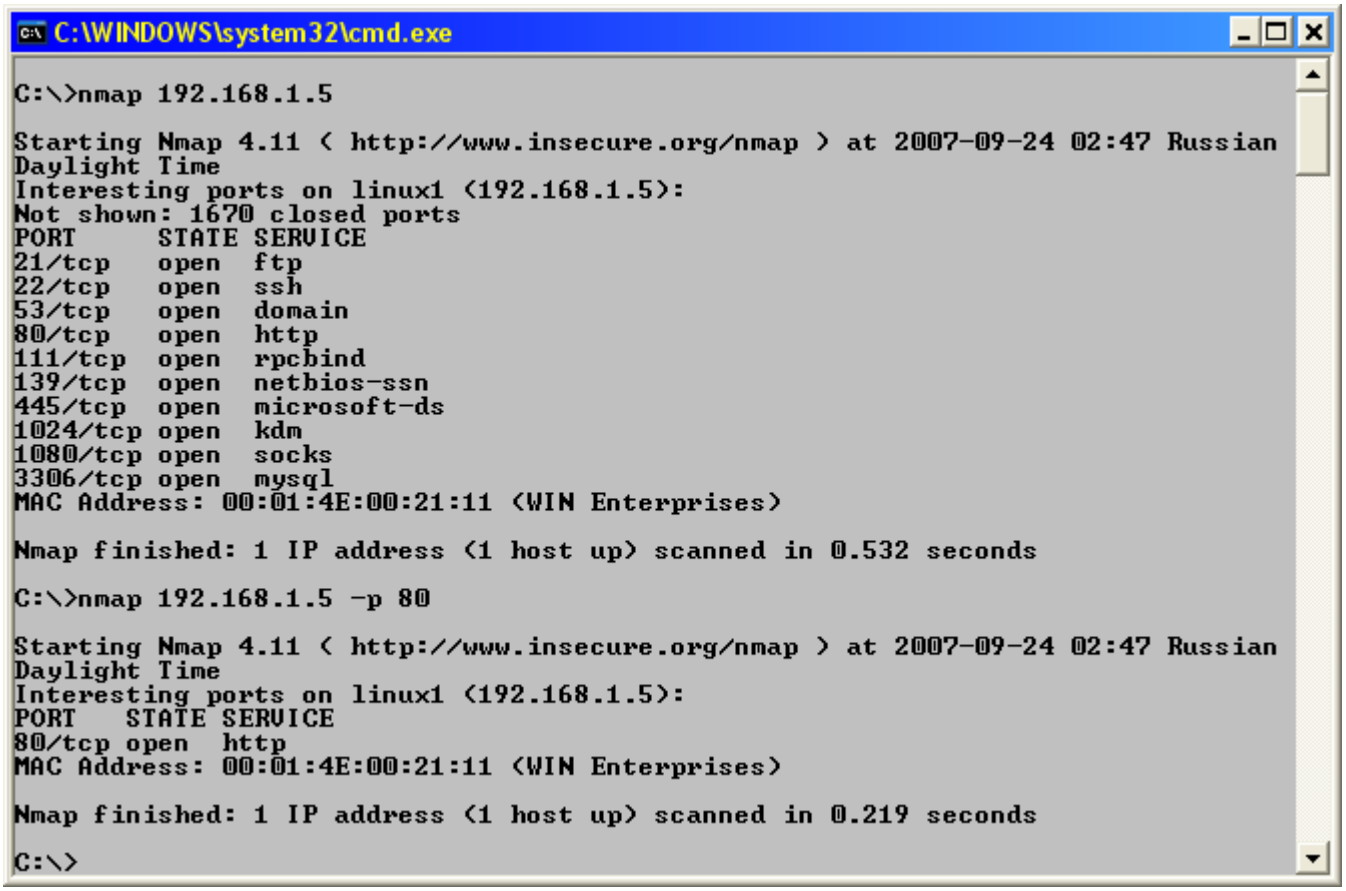


Рис. 2.11. Сканер портов ***nmap***

Сканер nmap портирован на большинство распространенных платформ: windows, linux, FreeBSD, OpenBSD и т.д.

Свежую информацию, статьи, бинарные файлы и исходные коды можно найти на официальном сайте http://insecure.org/nmap/

2. Порядок выполнения работы

1. Изучить стек протоколов TCP/IP.
2. Найти описание протоколов IP, TCP и UDP в соответствующих RFC.
3. Изучить утилиты netstat и tcpview: проанализировать текущие сетевые соединения на сетевой машине, получить статистику по протоколам (только netstat)
4. Изучить основные команды сканера портов nmap. Просканировать и зафиксировать сетевые службы соседнего компьютера.

3. Контрольные вопросы

1. Что такое сетевой протокол?
2. Зачем необходима стандартизация протоколов?
3. Понятие стека протоколов
4. Зачем введена модель OSI/ISO
5. Перечислите уровни стека протоколов TCP/IP и кратко охарактеризуйте их назначение.
6. Что такое IP-адрес?
7. В чем принципиальное отличие протоколов TCP и UDP.
8. Что такое сокет?
9. Зачем введен механизм сетевых портов?
10. Есть ли различие в протоколах реализованных, например, для ОС Windows и Linux?

ЛАБОРАТОРНАЯ РАБОТА № 3.

Разбиение на подсети

Цель работы: Изучить правила адресации сетевого уровня, научиться распределять адреса между участниками сети передачи данных .

1. Основные сведения

Сетевой уровень отвечает за возможность доставки пакетов по сети передачи данных – совокупности сегментов сети, объединенных в единую сеть любой сложности посредством узлов связи, в которой имеется возможность достижения из любой точки сети в любую другую.

В связи с необходимостью перенаправлять пакеты из одного сегмента сети в другой, сетевые адреса должны удовлетворять следующим требованиям:

- адреса должны быть *уникальны*. В сети не может быть нескольких участников с одинаковыми адресами во избежание неоднозначности
- сетевой адрес должен содержать информацию о том, *как достичь получателя* по сети.

Это приводит к структурности адреса – адрес разбивается на части, позволяющие определить местоположение участника внутри сети.

Структура может быть *сложной многоуровневой*, например адрес содержит информацию о стране, области, населенном пункте, предприятии, здании, отделе и т.д. или *простой*, содержащей номер сети и номер компьютера в сети.

По *сложной структуре* легче построить маршрут прохождения пакета, но адрес оказывается сложным и перегруженным часто ненужной информацией. Примером такой адресации может служить доменная адресация в Интернет, по адресу asu.bru.mogilev.by нетрудно понять, где находится данный участник сети и как до него добраться.

^ *Простая структура* позволяет значительно сократить размер адреса и сохраняет возможность работы в сети любой структуры, но для этого могут потребоваться сложные и, часто, не столь очевидные алгоритмы, как в предыдущем случае.

Протокол IP

Архитектуру сетевого уровня удобно рассматривать на примере сетевого протокола IP – самого распространенного в настоящее время, основного протокола сети Интернет. Термин «стек протоколов TCP/IP» означает «набор протоколов, связанных с IP и TCP(протоколом транспортного уровня)».

Архитектура протоколов TCP/IP предназначена для объединенной сети, состоящей из соединенных друг с другом шлюзами отдельных разнородных пакетных подсетей, к которым подключаются разнородные машины.

Каждая из подсетей работает в соответствии со своими специфическими требованиями и имеет свою природу средств связи. Однако предполагается, что каждая подсеть может принять пакет информации (данные с соответствующим сетевым заголовком) и доставить его по указанному

адресу в этой конкретной подсети.

^ Не требуется, чтобы подсеть гарантировала обязательную доставку пакетов и имела надежный сквозной протокол.

Таким образом, две машины, подключенные *к одной подсети*, могут обмениваться пакетами.

Когда необходимо передать пакет между машинами, *подключенными к разным подсетям*, то машина-отправитель посылает пакет в соответствующий шлюз (шлюз подключен к подсети также как обычный узел). Оттуда пакет направляется по определенному маршруту через систему шлюзов и подсетей, пока не достигнет шлюза, подключенного к той же подсети, что и машина-получатель: там пакет направляется к получателю.

Таким образом, адрес получателя должен содержать в себе:

- номер (адрес) подсети;
- номер (адрес) участника (хоста) внутри подсети.

IP адреса представляют собой 32-х разрядные двоичные числа. Для удобства их записывают в виде четырех десятичных чисел, разделенных точками. Каждое число является десятичным эквивалентом соответствующего байта адреса (для удобства будем записывать точки и в двоичном изображении).

192.168.200.47 является десятичным эквивалентом двоичного адреса

11000000.10101000.11001000.00101111

Иногда применяют десятичное значение IP-адреса. Его легко вычислить

$192*256^3+168*256^2+200*256+47=3232286767$

или с помощью метода Горнера :

$(((192*256)+168)*256+200)*256+47=3232286767$

Количество разрядов *адреса подсети* может быть различным и определяется *маской сети*.

^ *Маска сети* также является 32-х разрядным двоичным числом. Разряды маски имеют следующий смысл: если разряд маски равен 1, то соответствующий разряд адреса является разрядом адреса подсети, а если 0, то разрядом хоста внутри подсети. Все единичные разряды маски (если они есть) находятся в старшей (левой) части маски, а нулевые (если они есть) – в правой (младшей).

Исходя из вышесказанного, маску часто записывают в виде числа единиц в ней содержащихся.

255.255.248.0 (11111111.11111111.11111000.00000000) – является правильной маской подсети **(/21)**, а

255.255.250.0 (11111111.11111111.11111010.00000000) – является неправильной, недопустимой.

Нетрудно увидеть, что *максимальный размер подсети* может быть только степенью двойки (двойку надо возвести в степень, равную количеству нулей в маске).
При передаче пакетов используются *правила маршрутизации*, главное из которых звучит так: «Пакеты участникам своей подсети доставляются напрямую, а остальным – по другим правилам

маршрутизации».

Таким образом, требуется определить, является ли получатель членом нашей подсети или нет.

Маски при бесклассовой маршрутизации (CIDR)

Маски подсети являются основой метода бесклассовой маршрутизации (англ. CIDR). При этом подходе маску подсети записывают вместе с IP-адресом в формате «IP-адрес/количество единичных бит в маске». Число после знака дроби (т. н. длина префикса сети) означает количество единичных разрядов в маске подсети.

Рассмотрим пример записи диапазона IP-адресов в виде 10.96.0.0/11. В этом случае маска подсети будет иметь двоичный вид 11111111 11100000 00000000 00000000, или то же самое в десятичном виде: 255.224.0.0. 11 разрядов IP-адреса отводятся под адрес сети, а остальные 32-11=21 разряд полного адреса (11111111 11100000 00000000 00000000) — под локальный адрес в этой сети. Итого, 10.96.0.0/11 означает диапазон адресов от 10.96.0.0 до 10.127.255.255.

Таблица 3.1. **IPv4 CIDR**

IP/маска	До последнего IP в подсети	Маска	Количество адресов	Класс
a.b.c.d/**32**	+0.0.0.0	255.255.255.255	1	1/256 C
a.b.c.d/**31**	+0.0.0.1	255.255.255.254	2	1/128 C
a.b.c.d/**30**	+0.0.0.3	255.255.255.252	4	1/64 C
a.b.c.d/**29**	+0.0.0.7	255.255.255.248	8	1/32 C
a.b.c.d/**28**	+0.0.0.15	255.255.255.240	16	1/16 C
a.b.c.d/**27**	+0.0.0.31	255.255.255.224	32	1/8 C
a.b.c.d/**26**	+0.0.0.63	255.255.255.192	64	1/4 C
a.b.c.d/**25**	+0.0.0.127	255.255.255.128	128	1/2 C
a.b.c.0/**24**	+0.0.0.255	255.255.255.000	256	1 C
a.b.c.0/**23**	+0.0.1.255	255.255.254.000	512	2 C
a.b.c.0/**22**	+0.0.3.255	255.255.252.000	1024	4 C
a.b.c.0/**21**	+0.0.7.255	255.255.248.000	2048	8 C
a.b.c.0/**20**	+0.0.15.255	255.255.240.000	4096	16 C
a.b.c.0/**19**	+0.0.31.255	255.255.224.000	8192	32 C
a.b.c.0/**18**	+0.0.63.255	255.255.192.000	16 384	64 C
a.b.c.0/**17**	+0.0.127.255	255.255.128.000	32 768	128 C
a.b.0.0/**16**	+0.0.255.255	255.255.000.000	65 536	256 C = 1 B
a.b.0.0/**15**	+0.1.255.255	255.254.000.000	131 072	2 B
a.b.0.0/**14**	+0.3.255.255	255.252.000.000	262 144	4 B
a.b.0.0/**13**	+0.7.255.255	255.248.000.000	524 288	8 B
a.b.0.0/**12**	+0.15.255.255	255.240.000.000	1 048 576	16 B
a.b.0.0/**11**	+0.31.255.255	255.224.000.000	2 097 152	32 B
a.b.0.0/**10**	+0.63.255.255	255.192.000.000	4 194 304	64 B
a.b.0.0/**9**	+0.127.255.255	255.128.000.000	8 388 608	128 B
a.0.0.0/**8**	+0.255.255.255	255.000.000.000	16 777 216	256 B = 1 A

a.0.0.0/**7**	+1.255.255.255	254.000.000.000	33 554 432	2 A
a.0.0.0/**6**	+3.255.255.255	252.000.000.000	67 108 864	4 A
a.0.0.0/**5**	+7.255.255.255	248.000.000.000	134 217 728	8 A
a.0.0.0/**4**	+15.255.255.255	240.000.000.000	268 435 456	16 A
a.0.0.0/**3**	+31.255.255.255	224.000.000.000	536 870 912	32 A
a.0.0.0/**2**	+63.255.255.255	192.000.000.000	1 073 741 824	64 A
a.0.0.0/**1**	+127.255.255.255	128.000.000.000	2 147 483 648	128 A
0.0.0.0/**0**	+255.255.255.255	000.000.000.000	4 294 967 296	256 A

Определение диапазона адресов подсети.

Определение диапазона адресов подсети можно произвести из определения понятия маски:

- те разряды, которые относятся к адресу подсети, у всех хостов подсети должны быть*одинаковы*;
- адреса хостов в подсети могут быть *любыми*.

То есть, если наш адрес **192.168.200.47** и маска равна **/20,** то диапазон можно посчитать:

11000000.10101000.11001000.00101111 –адрес

11111111.11111111.11110000.00000000 –маска

11000000.10101000.1100XXXX.XXXXXXXX –диапазон адресов

где 0,1 – определенные значения разрядов,

X – любое значение,

Что приводит к диапазону адресов:

от **11000000.10101000.11000000.00000000 (192.168.192.0)**

до **11000000.10101000.11001111.11111111 (192.168.207.255)**

Следует учитывать, что некоторые адреса являются запрещенными или служебными и их нельзя использовать для адресов хостов или подсетей. Этоадреса, содержащие:

- **0** в *первом* или *последнем* байте,
- **255** в *любом* байте (это широковещательные адреса),
- **127** в *первом* байте (внутренняя петля – этот адрес имеется в каждом хосте и служит для связывания компонентов сетевого уровня).

Поэтому доступный диапазон адресов будет несколько меньше.

2. Порядок выполнения работы.

Используя схему сети, приведенную на следующем рисунке, а также информацию о количестве компьютеров в отделах предприятия, разбейте сеть на соответствующее количество подсетей. Разбиение должно быть оптимальным, то есть не следует использовать для отдела подсеть, если достаточно будет половины подсети. В отчете приведите:

1. схему сети с подписанными подсетями
2. параметры каждой подсети:

a. адрес сети (в двоичном и десятичном виде);
b. префикс;
c. маска (в двоичном и десятичном виде);
d. широковещательный адрес
e. адрес шлюза;
f. максимальное количество хостов;
g. количество неиспользуемых адресов хостов.

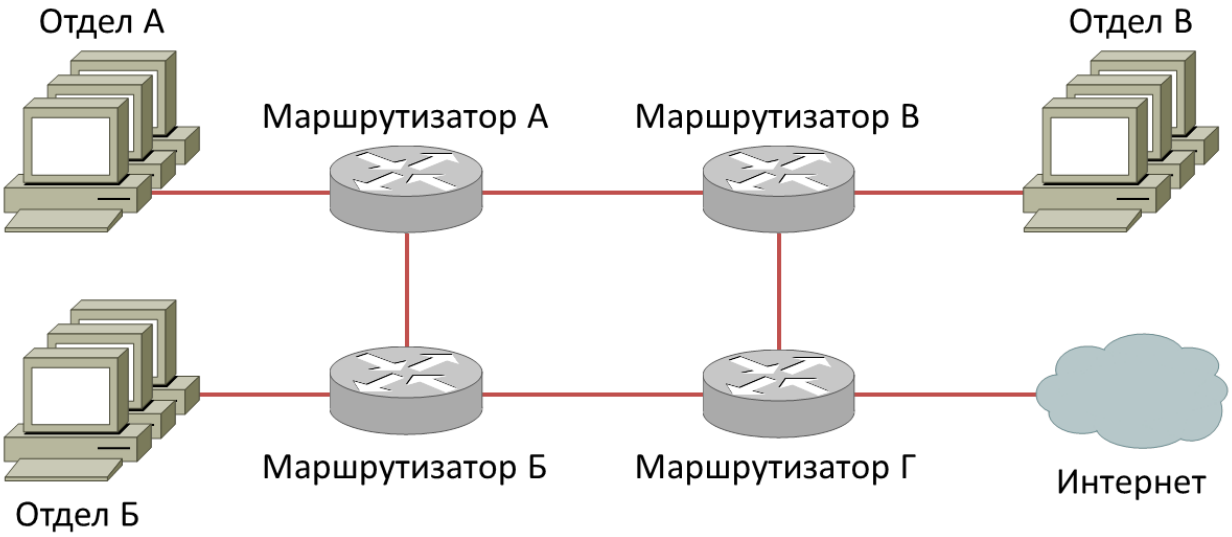


Рисунок 3.1. Схема сети предприятия

Таблица3. 2. Варианты заданий

№	Исходная сеть	Количество компьютеров в отделах		
		А	Б	В
1	34.178.0.0 /16	3750	6793	1702
2	118.7.50.0 /24	7	9	27
3	39.221.98.0 /24	8	5	18
4	88.27.252.0 /23	30	9	46
5	81.104.216.0 /21	48	120	249
6	7.50.128.0 /19	267	176	678
7	89.151.32.0 /19	311	246	806
8	126.61.74.0 /23	8	61	17
9	36.121.96.0 /19	311	696	226
10	28.54.64.0 /19	957	153	274
11	67.253.0.0 /16	3656	1165	5086
12	77.75.0.0 /18	338	830	1403
13	5.63.168.0 /21	119	61	226
14	85.123.72.0 /21	189	51	72
15	72.241.3.0 /25	12	7	3
16	87.228.68.0 /22	26	45	71
17	46.41.64.0 /18	384	1535	675
18	57.214.86.0 /23	63	9	21
19	74.30.128.0 /19	346	179	732
20	88.61.128.0 /20	366	77	130
21	10.58.180.0 /22	30	92	43
22	112.56.76.0 /22	23	114	60
23	2.78.160.0 /19	214	443	525
24	30.182.64.0 /18	624	1700	358
25	75.39.128.0 /19	625	219	372

3. Содержание отчета

1. титульный лист;
2. формулировку цели работы;
3. описание результатов выполнения;
4. выводы, согласованные с целью работы.

4. Контрольные вопросы.

1. Чем занимается сетевой уровень?
2. Что такое сеть передачи данных?
3. Какие требования предъявляются к сетевой адресации?
4. Можно ли использовать в качестве сетевого MAC-адрес?
5. Что такое маска подсети,?
6. Какова структура IP-адреса?
7. Чем определяется размер подсети?
8. Как определить диапазон адресов в подсети?
9. Как определить размер подсети?

ЛАБОРАТОРНАЯ РАБОТА № 4

Анализ пакетов локальной сети

Цель работы: Научиться пользоваться основными функциональными возможностями программы *Wireshark.*

Основные сведения

Wireshark представляет собой анализатор сетевого трафика с графическим интерфейсом.

Программа позволяет просматривать и анализировать пакеты, полученные из сетевого интерфейса или ранее собранного файла

1. Порядок выполнения работы:

В обычном режиме работы сетевая карта узла обрабатывает только те пакеты, которые адресованы данному узлу, или же всем узлам сети, если пакет широковещательный. Однако существует и другой режим работы сетевой карты – так называемый *неразборчивый режим* (Promiscuousmode), в котором сетевая карта получает все проходящие по сети пакеты.

Поскольку данный режим позволяет при определенных условиях перехватывать и анализировать трафик, предназначенный для чужих узлов, использование данного режима является незаконным в ряде стран, но даже если закон не запрещает его использование явно, перехват чужих данных может привести к возникновению административной или уголовной ответственности. Тем не менее, данный режим полезен в ряде случаев администраторам сети, поскольку позволяет отследить проблемный трафик (например, широковещательный шторм) и выяснить его причину.

Анализатор трафика (sniffer, сниффер) – это программа, предназначенная для перехвата и анализа сетевого трафика.

Последовательность выполнения работы

1) Запустите программу VirtualBox.
2) Выберите и запустите виртуальную машину Ubuntu.
3) Откройте терминал двойным щелчком по ярлыку на рабочем столе.
4) Введите команду sudo wireshark.
5) Узнайте пароль у преподавателя и введите его
6) В секции *Capture* в окне программы Wireshark выберите интерфейс eth0.
7) Запустите браузер FireFox и откройте какой-либо сайт, закройте браузер.
8) С помощью фильтра на панели инструментов отфильтруйте захваченный трафик, оставив только пакеты протокола TCP.
9) Скопируйте в отчёт строки анализатора трафика, относящиеся к следующим событиям (если за время захвата трафика было установлено несколько TCP-соединений, удалите из отчёта лишние строки):
   a) установка TCP-соединения;
   b) передача данных;
   c) завершение TCP-соединения.

10) Закройте Wireshark (на вопрос программы о сохранении результатов ответьте отрицательно) и другие открытые в виртуальной машине окна.
11) Завершите работу виртуальной машины: выберите в меню *Машина* пункт *Закрыть...*, отметьте пункт *Сохранить состояние машины* и нажмите кнопку ОК.
12) Дождитесь сохранения состояния виртуальной машины.
13) Закройте программу VirtualBox.

2. Содержание отчета

1) титульный лист;

2) формулировку цели работы;

3) описание результатов выполнения;

4) выводы, согласованные с целью работы.

3. Контрольные вопросы:

1. Предназначение анализаторов сетевого трафика.
2. Что такое сниффер?
3. Способы осуществления перехвата сетевого трафика.
4. Что может быть обнаружено в результате анализа сетевого трафика?
5. Наименование и функциональные возможности программ- анализаторов сетевого трафика.
6. Основные функциональные возможности программы *Wireshark.*
7. Отличия программы *Wireshark* от *tcpdump* .
8. Для чего применяется неразборчивый (англ. promiscuous mode) режим сетевой карты?

ЛАБОРАТОРНАЯ РАБОТА № 5.

Статическая маршрутизация

Цель: изучение процессов настройки статических маршрутов на маршрутизаторах Cisco.

1. Основные сведения

Подключение через терминал

Для административного подключения к маршрутизатору доступны несколько способов: через интернет, через телефонный канал, через терминал. Если маршрутизатор находится в физически доступном месте, то самым простым способом для подключения и конфигурирования будет использование терминала. В качестве эмулятора терминала можно использовать любую программу, которая обеспечивает подключение к устройству через COM-порт, например, `HyperTerminal`или `Putty`. Для физического подключения компьютера к маршрутизатору Ciscoиспользуется специальный консольный кабель Ciscoголубого цвета.

После запуска программы в настройках терминальной сессии следует установить ряд параметров, которые требуются для нормальной работы:

- Подключение: COM1
- Скорость бит: 9600
- Биты данных: 8
- Четность: нет
- Стоповые биты: 1
- Управление потоком: нет

Большинство эмуляторов терминала позволяют сохранить установки этих параметров и использовать их при последующих подключениях.

Режимы работы Cisco IOS

Для обеспечения безопасности Cisco IOS предоставляет четыре различных режима работы. Каждый из режимов позволяет выполнять определенные множества команд, и визуально отличается приглашением командной строки.

1. Пользовательский режим

Данный режим предоставляет ограниченные возможности, в нём разрешены лишь некоторые базовые операции для мониторинга. Он не позволяет выполнять никакие команды, которые могут изменить конфигурацию устройства. Приглашение командной строки в этом режиме выглядит следующим образом:

```
Router>
```

Вместо слова `Router` слева от знака режима может стоять другое имя, заданное пользователем.

2. Привилегированный режим

Выполнение конфигурационных или управляющих команд требует вхождения в привилегированный режим. Командная строка в этом режиме выглядит так:

```
Router#
```

Для перехода в привилегированный режим используется команда `enable`, для выхода – `disable`.

В рабочих сетях вход привилегированный режим может требовать авторизации пользователя. Изначально маршрутизаторы поставляются либо без пароля, либо с логином и паролем `cisco` / `cisco`.

3. Режим глобального конфигурирования

Из режима глобальной конфигурации можно делать изменения, который касаются устройства в целом. Также данный режим позволяет входить в режим конфигурирования определенного интерфейса. Приглашение командной строки в этом режиме выглядит так:

```
Router(config)#
```

Для входа в режим глобального конфигурирования используется команда `configure` с последующим указанием способа передачи конфигурационной информации. Например, если конфигурирование осуществляется с терминала, то команда будет выглядеть следующим образом:

```
Router#configureterminal
```

Вход в режим глобального конфигурирования возможен только из привилегированного режима.

4. Режим конфигурирования интерфейса

Данный режим представляет собой одно из подмножеств режима глобального конфигурирования и позволяет настраивать один из доступных сетевых интерфейсов (`Fa0/0`, `S0/0/0` и т. д.) Помимо режима конфигурирования интерфейса существуют также режимы конфигурирования линий (физических или виртуальных), а также режимы конфигурирования протоколов маршрутизатора. Все изменения, вносимые в конфигурацию маршрутизатора в данном режиме относятся только к выбранному интерфейсу. Приглашение командной строки в этом режиме выглядит следующим образом:

```
Router(config-if)#
```

Для входа в данный режим используется команда `interface`. Например, для конфигурирования интерфейса `FastEthernet 0/0` следует выполнить команду

```
Router(config)#interface fastethernet 0/0
```

КомандыCisco IOS

Cisco IOS предоставляет администратору широкий набор команд для управления работой маршрутизатора и его диагностики. При выборе команды следует помнить, что каждая из них может быть предназначена для работы только в определенном режиме, при попытке выполнить данную команду в другом режиме Cisco IOS её не опознает. Например, команда `hostname`, которая позволяет изменить внутреннее имя маршрутизатора, может быть выполнена только в режиме глобального конфигурирования:

```
Router>enable
Router#config t
Router(config)#hostname RtA
RtA(config)#
```

Действие ряда команд может быть отменено выполнением команды со специальной приставкой no, например, для отмены переименования маршрутизатора следует выполнить такую команду:

```
RtA(config)#no hostname
Router(config)#
```

Ещё одна важная команда, которая имеет множество дополнительных параметров и используется в разных режимах – это команда show. Например, чтобы посмотреть текущую конфигурацию маршрутизатора, следует выполнить команду

```
Router#showrunning-config
```

Cisco IOS отобразит ту конфигурацию, которая является актуальной в данный момент. Для просмотра конфигурации маршрутизатора, которая будет применена в случае перезапуска или сбоя питания, служит та же команда show, только с другим параметром:

```
Router#showstartup-config
```

Выводимая этой и другими командами информация может не помещаться в окне терминала. В этом случае Cisco IOS приостановит вывод и напечатает строчку

```
--More--
```

В этом случае следует нажать либо клавишу Пробел, которая выведет следующие строки на терминал, либо Enter, которая выведет только одну следующую строку. Нажатие комбинации Ctrl+C прервёт вывод текста и вернёт приглашение Cisco IOS.

Большинство команд Cisco IOS могут быть сокращены до нескольких первых букв: так, вместо configure terminal можно писать config t, а вместо fastethernet – просто fa.Сокращения ускоряют работу, однако усложняют понимание, поэтому не стоит чрезмерно ими увлекаться.

Конфигурация интерфейса

Как уже отмечалось выше, режим конфигурирования интерфейса позволяет управлять каждым интерфейсом независимо от остальных. Для входа в данный режим следует использовать команду interface в одном из следующих вариантов:

```
Router#interface тип порт
Router#interface тип слот/порт
Router#interface тип слот/подслот/порт
```

Обычно для физических интерфейсов обозначения типа, слота, подслота и порта написаны рядом с самим интерфейсом. Для выключения интерфейса служит команда shutdown, для повторного включения – команда noshutdown:

```
Router(config-if)#shutdown
Router(config-if)#no shutdown
```

Интерфейсу может быть назначен IP-адрес и маска сети:

```
Router(config-if)#ip address 192.168.1.1 255.255.255.0
```

Если интерфейсы двух маршрутизаторов соединены между собой, то IP-адреса каждого из интерфейсов должны находиться в одной подсети!

Для выхода из режима конфигурации интерфейса используется команда `exit`:

```
Router(config-if)#exit
Router(config)#
```

Команды мониторинга

Для проверки связи между устройствами сети можно использовать команды `ping` и `traceroute`:

```
Router#ping 192.168.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 192.168.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

Каждый ICMP-пакет, на который был получен ответ, обозначается восклицательным знаком, каждый потерянный пакет – точкой.

```
Router#traceroute 192.168.1.2
Type escape sequence to abort.
Tracing the route to 192.168.1.2
  1 172.16.0.253 8 msec 4 msec 8 msec
2 10.0.0.254 16 msec 16 msec 8 msec
  3 192.168.1.1 16 msec * 20 msec
```

Для проверки состояния интерфейса можно использовать команду `show` с дополнительными параметрами:

```
Router#show ip interface brief
Interface        IP-Address  OK?  Meth.  Status Protocol
FastEthernet0/0  192.168.1.1 YES  NVRAM  up         up
Serial0/0/1      unassigned  YES  unset  adm. down  down
```

Глядя на вывод данной команды (часть текста опущена) можно определить, что интерфейсу `FastEthernet0/0` назначен IP-адрес `192.168.1.1`, последние две колонки определяют статус интерфейса – у включенного работоспособного интерфейса оба параметра должны иметь значение `up`. Значение `administratively down` означает, что интерфейс был отключен администратором.

Статическая маршрутизация

Статическая маршрутизация предполагает фиксированную структуру сети: каждый маршрутизатор в сети точно знает куда нужно отправлять пакет, чтобы он был доставлен по назначению. С одной стороны, это упрощает работу администратора в небольших сетях –

достаточноодин раз прописать все маршруты; такая конфигурация не будет создавать дополнительный служебный трафик внутри сети. С другой стороны, если сеть достаточно большая, то ручная настройка всех маршрутизаторов является утомительным делом, не исключены и ошибки. Более того, при изменении топологии сети нужно будет вручную переконфигурировать каждое из сетевых устройств.

Рассмотрим статическую маршрутизацию на примере. Если два маршрутизатора соединены между собой, то каждый из них знает о том, какая сеть подключена непосредственно к нему:

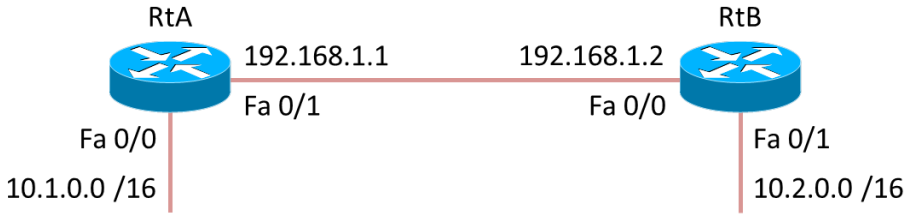


Рис.5. 1. Сеть из двух маршрутизаторов.

Так, маршрутизатор `RtA` знает, что для отправки пакета в сеть `10.1.0.0/16` нужно использовать интерфейс `Fa 0/0`, а сеть `192.168.1.0/24` подключена непосредственно к интерфейсу `Fa 0/1`, однако он ничего не знает о сети `10.2.0.0/16`. Для того, чтобы пакеты могли быть доставлены в правильную сеть, нужно сообщить маршрутизатору `RtA`, что они должны быть отправлены на маршрутизатор `RtB`, который, в свою очередь, знает о сети `RtB`. Делается это с помощью команды `ip route`, в качестве параметра которой указывается либо соответствующий *выходной* интерфейс текущего маршрутизатора, либо адрес *входного* интерфейса маршрутизатора назначения. Таким образом, обе следующие команды эквивалентны:

```
RtA(config)#iproute 10.2.0.0 255.255.0.0 192.168.1.2
RtA(config)#ip route 10.2.0.0 255.255.0.0 Fa0/1
```

Однако если теперь попробовать из сети `10.1.0.0/16` выполнить команду `ping` с адресом назначения в сети `10.2.0.0/16`, то результаты по-прежнему будут неудовлетворительными. Это происходит потому, что пакеты в сеть назначения доставляются, но ответы обратно не доходят, ведь маршрутизатор `RtB` ничего не знает о сети `10.1.0.0/16`, поэтому для маршрутизатора `RtB` также необходимо прописать статические маршруты.

Для хранения информации об известных маршрутах маршрутизатор использует таблицу маршрутизации. Просмотреть её можно с помощью следующей команды:

```
RtA#showiproute
Codes: C - connected, S - static, I - IGRP, R - RIP
B – BGP, D - EIGRP, EX - EIGRP external, O – OSPF
IA - OSPF inter area, N1 - OSPF NSSA external type 1
N2 - OSPF NSSA external type 2, E1 - OSPF external type
E2 - OSPF external type 2, E – EGP, i – IS-IS
L1 - IS-IS level-1, L2 - IS-IS level-2,
ia - IS-IS inter area, * - candidate default
U - per-user static route, o - ODR
P - periodic downloaded static route

Gateway of last resort is not set

S   192.168.1.0/24 is directly connected, FastEthernet0/1
C   192.168.2.0/24 is directly connected, FastEthernet0/1
C   192.168.3.0/24 is directly connected, FastEthernet0/0
```

В первой части вывода этой команды приведена легенда – обозначения кодов каждой записи из таблицы маршрутизатора. Например, код `C` означает, что данная сеть непосредственно подключена к одному из интерфейсов, код `S` – что данный маршрут был добавлен вручную (поэтому он называется статическим), код `R` – что данный маршрут был получен в результате работы протокола RIP и т.д. Ниже идут непосредственно сами маршруты: для каждого из них указана сеть назначения, количество бит сети в адресе, а также интерфейс, в который будет отправлен пакет для доставки в нужную сеть.

2. Порядок выполнения работы

1. Запустите Cisco Packet Tracer соединить и настроить маршрутизаторы для работы в сети со следующей топологией.

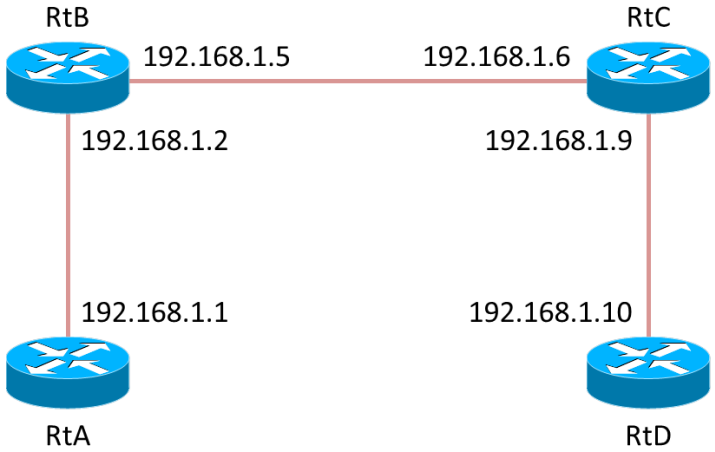


Рис. 2.1. Сеть из четырёх маршрутизаторов.

2. Соедините соответствующие порты маршрутизаторов перекрестными кабелями.
3. Запустите терминальную программу, например, HyperTerminal и откройте терминальную сессию с нужными параметрами.
4. Подключите консольный кабель к первому маршрутизатору.
5. Пользуясь терминалом:
   a) войдите в режим глобальной конфигурации;
   b) измените имя маршрутизатора на `RtA`;
   c) настройте интерфейс, к которому подключен соседний маршрутизатор:
      i. войдите в режим конфигурирования интерфейса;
      ii. задайте IP-адрес для данного интерфейса;
      iii. активируйте интерфейс;
      iv. выйдите из режима конфигурирования интерфейса;
   d) если у маршрутизатора используются другие интерфейсы, то повторите шаг с для каждого из них;
   e) пропишите статические пути для каждой сети, которая не является соседней для данного маршрутизатора;
   f) выйдите из режима глобальной конфигурации;
6. Повторите пункты 3-4 для каждого маршрутизатора.
7. Выполните проверку связи между маршрутизаторами `RtA` и `RtD` в обоих направлениях с помощью команд `ping` и `traceroute`.
8. В отчете отразите следующую информацию по каждому маршрутизатору:
   a) команды, необходимые для конфигурации, с пояснениями сути каждой команды;
   b) таблицу маршрутизации;
   c) результаты выполнения команд `ping` и `traceroute`.

2. Содержание отчета

1) титульный лист;

2) формулировку цели работы;

3) описание результатов выполнения;

4) выводы, согласованные с целью работы.

ЛАБОРАТОРНАЯ РАБОТА №6

Изучение утилит TCP/IP в ОС Windows

Цель: научиться пользоваться диагностическими утилитами, предназначенными для проверки конфигурации стека и тестирования сетевого соединения

1. Основные сведения

Диагностические утилиты TCP/IP

В состав TCP/IP входят диагностические утилиты, предназначенные для проверки конфигурации стека и тестирования сетевого соединения.

Утилита	**Применение**
hostname	Выводит имя локального хоста. Используется без параметров.
ipconfig	Выводит значения для текущей конфигурации стека TCP/IP: IP-адрес, маску подсети, адрес шлюза по умолчанию, адреса WINS (Windows Internet Naming Service) и DNS (Domain Name System)
ping	Осуществляет проверку правильности конфигурирования TCP/IP и проверку связи с удаленным хостом.
tracert	Осуществляет проверку маршрута к удаленному компьютеру путем отправки эхо-пакетов протокола ICMP (Internet Control Message Protocol). Выводит маршрут прохождения пакетов на удаленный компьютер.
arp	Выводит для просмотра и изменения таблицу трансляции адресов, используемую протоколом разрешения адресов ARP (Address Resolution Protocol - определяет локальный адрес по IP-адресу)
route	Модифицирует таблицы маршрутизации IP. Отображает содержимое таблицы, добавляет и удаляет маршруты IP.
netstat	Выводит статистику и текущую информацию по соединению TCP/IP.
nslookup	Осуществляет проверку записей и доменных псевдонимов хостов, доменных сервисов хостов, а также информации операционной системы, путем запросов к серверам DNS.

Проверка правильности конфигурации TCP/IP с помощью ipconfig.

При устранении неисправностей и проблем в сети TCP/IP следует сначала проверить правильность конфигурации TCP/IP. Для этого используется утилита ipconfig.

Эта команда полезна на компьютерах, работающих с DHCP (Dynamic Host Configuration Protocol), так как дает пользователям возможность определить, какая конфигурация сети TCP/IP и какие величины были установлены с помощью DHCP.

Синтаксис:

ipconfig [/all | /renew[adapter] | /release]

Параметры:

all	выдает весь список параметров. Без этого ключа отображается только IP-адрес, маска и шлюз по умолчанию;
renew[adapter]	обновляет параметры конфигурации DHCP для указанного сетевого адаптера;
release[adapter]	освобождает выделенный DHCP IP-адрес; adapter – имя сетевого адаптера;
displaydns	выводит информацию о содержимом локального кэша клиента DNS, используемого для разрешения доменных имен.

Таким образом, утилита ipconfig позволяет выяснить, инициализирована ли конфигурация и не дублируются ли IP-адреса:

- если конфигурация инициализирована, то появляется IP-адрес, маска, шлюз;
- если IP-адреса дублируются, то маска сети будет 0.0.0.0;
- если при использовании DHCP компьютер не смог получить IP-адрес, то он будет равен 0.0.0.0 .

Тестирование связи с использованием утилиты *ping*.

Утилита ping (Packet Internet Grouper) используется для проверки конфигурирования TCP/IP и диагностики ошибок соединения. Она определяет доступность и функционирование конкретного хоста. Использование ping лучший способ проверки того, что между локальным компьютером и сетевым хостом существует маршрут. Хостом называется любое сетевое устройство (компьютер, маршрутизатор), обменивающееся информацией с другими сетевыми устройствами по TCP/IP.

Команда ping проверяет соединение с удаленным хостом путем отправки к этому хосту эхо-пакетов ICMP и прослушивания эхо-ответов. Ping ожидает каждый посланный пакет и печатает количество переданных и принятых пакетов. Каждый принятый пакет проверяется в соответствии с переданным сообщением. Если связь между хостами плохая, из сообщений ping станет ясно, сколько пакетов потеряно.

По умолчанию передается 4 эхо-пакета длиной 32 байта (возможны и другие варианты значения по умолчанию) - периодическая последовательность символов алфавита в верхнем регистре. Ping позволяет изменить размер и количество пакетов, указать, следует ли записывать маршрут, который она использует, какую величину времени жизни (ttl) устанавливать, можно ли фрагментировать пакет и т.д.. При получении ответа в поле time указывается, за какое время (в миллисекундах) отправленный пакет доходит до удаленного хоста и возвращается назад. Так как значение по умолчанию для ожидания отклика равно 1 секунде, то все значения данного поля будут меньше 1000 миллисекунд. Если вы получаете сообщение «Request time out» (Превышен интервал ожидания), то, возможно, если увеличить время ожидания отклика, пакет дойдет до удаленного хоста. Это можно сделать с помощью ключа –w.

Ping можно использовать для тестирования как имени хоста (DNS или NetBIOS), так и его IP-адреса. Если ping с IP-адресом выполнилась успешно, а с именем – неудачно, это значит, что проблема заключается в распознавании соответствия адреса и имени, а не в сетевом соединении.

Утилита ping используется следующими способами:

1) Для проверки того, что TCP/IP установлен и правильно сконфигурирован на локальном компьютере, в команде ping задается адрес петли обратной связи (loopback address):

ping 127.0.0.1

Если тест успешно пройден, то вы получите следующий ответ:

Ответ от 127.0.0.1: число байт=32 время<1мс TTL=128

Ответ от 127.0.0.1: число байт=32 время<1мс TTL=128

Ответ от 127.0.0.1: число байт=32 время<1мс TTL=128

Ответ от 127.0.0.1: число байт=32 время<1мс TTL=128

2) Чтобы убедиться в том, что компьютер правильно добавлен в сеть и IP-адрес не дублируется, используется IP-адрес локального компьютера:

ping IP-адрес_локального_хоста

3) Чтобы проверить, что шлюз по умолчанию функционирует и что можно установить соединение с любым локальным хостом в локальной сети, задается IP-адрес шлюза по умолчанию:

ping IP-адрес_шлюза

4) Для проверки возможности установления соединения через маршрутизатор в команде ping задается IP-адрес удаленного хоста:

ping IP-адрес_удаленного хоста

Синтаксис:

ping [-t] [-a] [-n count] [-l length] [-f] [-i ttl] [-v tos] [-r count] [-s count] [[-j host-list] |
[-k host-list]] [-w timeout] destination-list

Параметры:

-t выполняет команду ping до прерывания. Control-Break - посмотреть статистику и продолжить. Control-C - прервать выполнение команды;

-a позволяет определить доменное имя удаленного компьютера по его IP-адресу;

-n count посылает количество пакетов ECHO, указанное параметром count;

-l length посылает пакеты длиной length байт (максимальная длина 8192 байта);

-f посылает пакет с установленным флагом «не фрагментировать». Этот пакет не будет фрагментироваться на маршрутизаторах по пути своего следования;

-i ttl устанавливает время жизни пакета в величину ttl (каждый маршрутизатор уменьшает ttl на единицу);

-v tos устанавливает тип поля «сервис» в величину tos;

-r count записывает путь выходящего пакета и возвращающегося пакета в поле записи пути. Count - от 1 до 9 хостов;

-s count позволяет ограничить количество переходов из одной подсети в другую (хопов). Count задает максимально возможное количество хопов;

-j host-list направляет пакеты с помощью списка хостов, определенного параметром host-list. Последовательные хосты могут быть отделены промежуточными маршрутизаторами (гибкая статическая маршрутизация). Максимальное количество хостов в списке, позволенное IP, равно 9;

-k host-list направляет пакеты через список хостов, определенный в host-list. Последовательные хосты не могут быть разделены промежуточными маршрутизаторами (жесткая статическая маршрутизация). Максимальное количество хостов – 9;

-w timeout указывает время ожидания (timeout) ответа от удаленного хоста в миллисекундах (по умолчанию – 1сек);

destination-list указывает удаленный хост, к которому надо направить пакеты ping.

<u>Пример использования утилиты ping:</u>

C:\WINDOWS>ping –n 10 www.netscape.com

Обмен пакетами с www.netscape.com [205.188.247.65] по 32 байт:

Ответ от 205.188.247.65: число байт=32 время=194мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=240мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=173мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=250мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=187мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=239мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=263мс TTL=48

Ответ от 205.188.247.65: число байт=32 время=230мс TTL=48
Ответ от 205.188.247.65: число байт=32 время=185мс TTL=48
Ответ от 205.188.247.65: число байт=32 время=406мс TTL=48
Статистика Ping для 205.188.247.65:
Пакетов: послано = 10, получено = 10, потеряно = 0 (0% потерь)
Приблизительное время передачи и приема:
Наименьшее = 173мс, наибольшее = 406мс, среднее =236мс

В случае невозможности проверить доступность хоста утилита выводит информацию об ошибке. Ниже приведен пример ответа утилиты ping при попытке послать запрос на несуществующий хост.

Обмен пакетами с 172.16.6.21 по 32 байт:

Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.

Статистика Ping для 172.16.6.21:
Пакетов: отправлено = 4, получено = 0, потеряно = 4 (100% потерь),

Приблизительное время передачи и приема:
наименьшее = 0мс, наибольшее = 0мс, среднее = 0мс

Утилита сообщает не об отсутствии хоста, а о том, что за отведенное время не был получен ответ на посланный запрос. Причиной этого не обязательно является отсутствие хоста в сети. Проблема может крыться в сбоях связи, перегрузке или неправильной настройке маршрутизаторов и т. п. Ошибка «сеть недоступна» (network unreachable) прямо указывает на проблемы маршрутизации.

Изучение маршрута между сетевыми соединениями с помощью утилиты *tracert*.

Tracert - это утилита трассировки маршрута. Она использует поле TTL (time-to-live, время жизни) пакета IP и сообщения об ошибках ICMP для определения маршрута от одного хоста до другого.

Утилита tracert может быть более содержательной и удобной, чем ping, особенно в тех случаях, когда удаленный хост недостижим. С помощью нее можно определить район проблем со связью (у Internet-провайдера, в опорной сети, в сети удаленного хоста) по тому, насколько далеко будет отслежен маршрут. Если возникли проблемы, то утилита выводит на экран звездочки (*), либо сообщения типа «Destination net unreachable», «Destination host unreachable», «Request time out», «Time Exeeded».

Утилита tracert работает следующим образом: посылается по 3 пробных эхо-пакета на каждый хост, через который проходит маршрут до удаленного хоста. На экран при этом выводится время ожидания ответа на каждый пакет (Его можно изменить с помощью параметра -w). Пакеты посылаются с различными величинами времени жизни. Каждый маршрутизатор, встречающийся по пути, перед перенаправлением пакета уменьшает величину TTL на единицу. Таким образом, время жизни является счетчиком точек промежуточной доставки (хопов). Когда время жизни пакета достигнет нуля, предполагается, что маршрутизатор пошлет в компьютер-источник сообщение ICMP “Time Exeeded” (Время истекло). Маршрут определяется путем посылки первого эхо-пакета с TTL=1. Затем TTL увеличивается на 1 в каждом последующем пакете до тех пор, пока пакет не достигнет удаленного хоста, либо будет достигнута максимально возможная величина TTL (по умолчанию 30, задается с помощью параметра -h).

Маршрут определяется путем изучения сообщений ICMP, которые присылаются обратно промежуточными маршрутизаторами.

Примечание: некоторые маршрутизаторы просто уничтожают пакеты с истекшим TTL и не будут видны утилите tracert.

Синтаксис:

tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout] имя_целевого_хоста

Параметры:

-d указывает, что не нужно распознавать адреса для имен хостов;

-h maximum_hops указывает максимальное число хопов для того, чтобы искать цель;

-j host-list указывает нежесткую статическую маршрутизацию в соответствии с host-list;

-w timeout указывает, что нужно ожидать ответ на каждый эхо-пакет заданное число мсек.

Утилита *arp*.

Основная задача протокола ARP – трансляция IP-адресов в соответствующие локальные адреса. Для этого ARP-протокол использует информацию из ARP-таблицы (ARP-кэша). Если необходимая запись в таблице не найдена, то протокол ARP отправляет широковещательный запрос ко всем компьютерам локальной подсети, пытаясь найти владельца данного IP-адреса. В

кэше могут содержаться два типа записей: статические и динамические. Статические записи вводятся вручную и хранятся в кэше постоянно. Динамические записи помещаются в кэш в результате выполнения широковещательных запросов. Для них существует понятие времени жизни. Если в течение определенного времени (по умолчанию 2 мин.) запись не была востребована, то она удаляется из кэша.

Синтаксис:

arp [-s inet_addr eth_addr] | [-d inet_addr] | [-a]

Параметры:

-s занесение в кэш статических записей;

-d удаление из кэша записи для определенного IP-адреса;

-a просмотр содержимого кэша для всех сетевых адаптеров локального компьютера;

inet_addr - IP-адрес;

eth_addr - MAC-адрес.

Утилита route.

Утилита **route** предназначена для работы с локальной таблицей маршрутизации. Она имеет следующий **синтаксис:**

route [-f] [-p] [команда [узел] [MASK маска] [шлюз] [METRIC метрика] [IF интерфейс]]

Параметры:

-f Очистка таблицы маршрутизации.

-p При указании совместно с командой ADD создает постоянную запись, которая сохраняется после перезагрузки компьютера. По умолчанию записи таблицы маршрутов не сохраняются при перезагрузке.

команда одна из четырех команд:

PRINT - вывод информации о маршруте;

ADD - добавление маршрута;

DELETE - удаление маршрута;

CHANGE - изменение маршрута.

узел адресуемый узел

маска маска подсети; по умолчанию используется маска 255.255.255.255

шлюз адрес шлюза

метрика метрика маршрута;

интерфейс идентификатор интерфейса, который будет использован для пересылки пакета

Для команд PRINT и DELETE возможно использование символов подстановки при указании адресуемого узла или шлюза. Параметр шлюза для этих команд может быть опущен. При добавлении и изменении маршрутов утилита route осуществляет проверку введенной информации на соответствие условию (УЗЕЛ & МАСКА) == УЗЕЛ. Если это условие не выполняется, то утилита выдает сообщение об ошибке и не добавляет или не изменяет маршрут.
Утилита осуществляет поиск имен сетей в файле networks. Поиск имен шлюзов осуществляется в файле hosts. Оба файла расположены в папке %systemroot%\system32\drivers\etc. Наличие и заполнение этих файлов не обязательно для нормального функционирования утилиты route и работы маршрутизации.
Хотя в большинстве случаев на рабочей станции это не требуется, можно вручную редактировать таблицы маршрутизации.

Пример использования утилиты route:
Добавление статического маршрута:
route add 172.16.6.0 MASK 255.255.255.0 172.16.11.1 METRIC 1 IF 0x1000003

Утилита *netstat.*

Утилита netstat позволяет получить статическую информацию по некоторым из протоколов стека (TCP, UDP, IP, ICMP), а также выводит сведения о текущих сетевых соединениях. Особенно она полезна на брандмауэрах, с ее помощью можно обнаружить нарушения безопасности периметра сети.

Синтаксис:

netstat [-a] [-e] [-n] [-s] [-p protocol] [-r]

Параметры:

-a выводит перечень всех сетевых соединений и прослушивающихся портов локального компьютера;

-e выводит статистику для Ethernet-интерфейсов (например, количество полученных и отправленных байт);

-n выводит информацию по всем текущим соединениям (например, TCP) для всех сетевых интерфейсов локального компьютера. Для каждого соединения выводится информация об IP-адресах локального и удаленного интерфейсов вместе с номерами используемых портов;

-s выводит статистическую информацию для протоколов UDP, TCP, ICMP, IP. Ключ «/more» позволяет просмотреть информацию постранично;

-r выводит содержимое таблицы маршрутизации.

Утилита *nslookup*.

Утилита **nslookup** предназначена для диагностики службы DNS, в простейшем случае - для выполнения запросов к DNS-серверам на разрешение имен в IP-адреса. В общем случае утилита позволяет просмотреть любые записи DNS-сервера:

A – каноническое имя узла, устанавливает соответствие доменного имени ip-адресу.

SOA – начало полномочий, начальная запись, единственная для зоны;

MX – почтовые серверы (хосты, принимающие почту для заданного домена);

NS – серверы имен (содержит авторитетные DNS-серверы для зоны);

PTR – указатель (служит для обратного преобразования ip-адреса в символьное имя хоста)

и т. д.

Утилита nslookup достаточно сложна и содержит свой собственный командный интерпретатор.

В простейшем случае (без входа в командный режим) утилита **nslookup** имеет следующий

Синтаксис:

nslookup хост [сервер]

Параметры:

Хост DNS-имя хоста, которое должно быть преобразовано в IP-адрес.

Сервер Адрес DNS-сервера, который будет использоваться для разрешения имени. Если этот параметр опущен, то будут последовательно использованы адреса DNS-серверов из параметров настройки протокола TCP/IP.

Примеры использования утилиты nslookup:

1. Получение списка серверов имен для домена yandex.ru без входа в командный режим (с использованием ключей).

```
C:\> nslookup -type=ns yandex.ru
Server:  dns01.catv.ext.ru
Address:  217.10.44.35
Non-authoritative answer:
yandex.ru       nameserver = ns4.yandex.ru
yandex.ru       nameserver = ns5.yandex.ru
yandex.ru       nameserver = ns2.yandex.ru
yandex.ru       nameserver = ns1.yandex.ru
```

ns2.yandex.ru internet address = 213.180.199.34

ns5.yandex.ru internet address = 213.180.204.1

2. Получение записи SOA домена yandex.ru с авторитетного сервера с использование командного интерпретатора nslookup.

C:\>nslookup

Default Server: dns04.catv.ext.ru

Address: 217.10.39.4

> set type=SOA

> server ns2.yandex.ru

Default Server: ns2.yandex.ru

Address: 213.180.199.34

> yandex.ru

Server: ns1.yandex.ru

Address: 213.180.193.1

>yandex.ru

    primary name server = ns1.yandex.ru

    responsible mail addr = sysadmin.yandex-team.r

    serial = 2009022707

    refresh = 1800 (30 mins)

    retry   = 900 (15 mins)

    expire = 2592000 (30 days)

    default TTL = 900 (15 mins)

yandex.ru nameserver = ns5.yandex.ru

yandex.ru nameserver = ns1.yandex.ru

yandex.ru nameserver = ns4.yandex.ru

yandex.ru nameserver = ns2.yandex.ru

ns1.yandex.ru internet address = 213.180.193.1

ns2.yandex.ru internet address = 213.180.199.34

ns4.yandex.ru internet address = 77.88.19.60

ns5.yandex.ru internet address = 213.180.204.1

> exit

3. Получение адреса почтового сервера для домена yandex.ru.

C:\ >nslookup

Default Server: dns01.catv.ext.ru

Address: 217.10.44.35

> set q=mx

> yandex.ru

Server: dns01.catv.ext.ru

Address: 217.10.44.35

Non-authoritative answer:

yandex.ru MX preference = 10, mail exchanger = mx2.yandex.ru

yandex.ru MX preference = 10, mail exchanger = mx3.yandex.ru

yandex.ru MX preference = 10, mail exchanger = mx1.yandex.ru

yandex.ru nameserver = ns2.yandex.ru

yandex.ru nameserver = ns1.yandex.ru

yandex.ru nameserver = ns4.yandex.ru

yandex.ru nameserver = ns5.yandex.ru

mx1.yandex.ru internet address = 77.88.21.89

mx2.yandex.ru internet address = 93.158.134.89

mx3.yandex.ru internet address = 213.180.204.89

ns2.yandex.ru internet address = 213.180.199.34

ns4.yandex.ru internet address = 77.88.19.60

ns5.yandex.ru internet address = 213.180.204.1

>

Указав ключ type=any, можно получить все записи о узле или домене. Ключи querytype, t, q эквивалентны type.

2. Порядок выполнения работы

1. Выведите на экран справочную информацию по всем рассмотренным утилитам (см. таблицу п.1). Для этого в командной строке введите имя утилиты без параметров и дополните **/?.**

Сохраните справочную информацию в отдельном файле.

Изучите ключи, используемые при запуске утилит.

2.Выведите на экран имя локального хоста с помощью команды hostname. Сохраните результат в отдельном файле.

3.Проверьте конфигурацию TCP/IP с помощью утилиты ipconfig. Заполните таблицу:

Имя хоста	
IP-адрес	
Маска подсети	

Основной шлюз	
Используется ли DHCP (адрес DHCP-сервера)	
Описание адаптера	
Физический адрес сетевого адаптера	
Адрес DNS-сервера	
Адрес WINS-сервера	

4. Проверьте правильность установки и конфигурирования TCP/IP на локальном компьютере.

- Проверьте функционирование основного шлюза, послав 5 эхо-пакетов длиной 64 байта.
- Проверьте возможность установления соединения с удаленным хостом.
- С помощью команды ping проверьте адреса a) yandex.ru; b) madi.ru; c) google.com и для каждого из них отметьте время отклика. Попробуйте изменить параметры команды ping таким образом, чтобы увеличилось время отклика. Определите IP-адреса узлов.

4. С помощью команды tracert проверьте для перечисленных ниже адресов, через какие промежуточные узлы идет сигнал. Изучите ключи команды.

a) madi21.ru

b) mathmod.aspu.ru

c) yarus.aspu.ru

5. С помощью утилиты arp просмотрите ARP-таблицу локального компьютера.

Внести в кэш локального компьютера любую статическую запись.

7. С помощью утилиты route просмотреть локальную таблицу маршрутизации.

8. С помощью утилиты netstat выведите перечень сетевых соединений и статистическую информацию для протоколов UDP, TCP, ICMP, IP.

3. Контрольные вопросы

1. Раскрыть термины: хост, шлюз, хоп, время жизни пакета, маршрут, маска сети, авторитетный/неавторитетный (компетентный) DNS-сервер, порт TCP, петля обратной связи, время отклика.

2. Какие утилиты можно использовать для проверки правильности конфигурирования TCP/IP?
3. Каким образом команда ping проверяет соединение с удаленным хостом?
4. Каково назначение протокола ARP?
5. Как утилита ping разрешает имена узлов в ip-адреса (и наоборот)?
6. Какие могут быть причины неудачного завершения ping и tracert? (превышен интервал ожидания для запроса, сеть недоступна, превышен срок жизни при передаче пакета).
7. Всегда ли можно узнать символьное имя узла по его ip-адресу?
8. Какой тип записи запрашивает у DNS-сервера простейшая форма nslookup?

4. Содержание отчета

1) титульный лист;

2) формулировку цели работы;

3) описание результатов выполнения;

4) выводы, согласованные с целью работы.

ЛАБОРАТОРНАЯ РАБОТА №7

ACL: списки контроля доступа

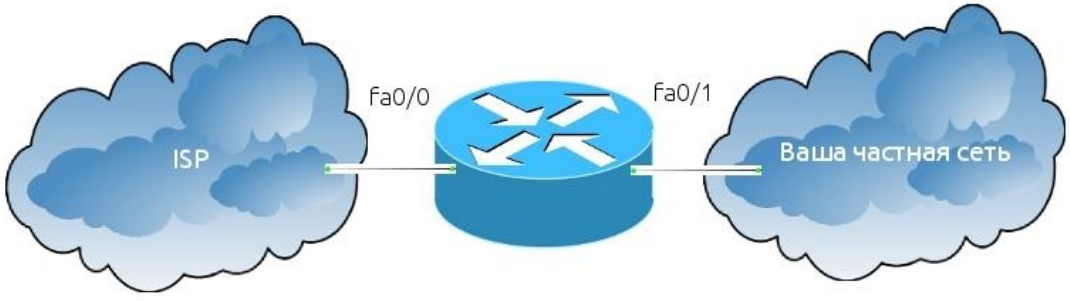
Цель работы: Научиться разрабатывать собственные списки контроля доступа .

1. Общие сведения

ACL (Access Control List) — это набор текстовых выражений, которые что-то разрешают, либо что-то запрещают. Обычно ACL разрешает или запрещает IP-пакеты, но помимо всего прочего он может заглядывать внутрь IP-пакета, просматривать тип пакета, TCP и UDP порты. Также ACL существует для различных сетевых протоколов (IP, IPX, AppleTalk и так далее). В основном применение списков доступа рассматривают с точки зрения пакетной фильтрации, то есть пакетная фильтрация необходима в тех ситуациях, когда у вас стоит оборудование на границе Интернет и вашей частной сети и нужно отфильтровать ненужный трафик.
Вы размещаете ACL на входящем направлении и блокируете избыточные виды трафика.
Функционал ACL состоит в классификации трафика, нужно его проверить сначала, а потом что-то с ним сделать в зависимости от того, куда ACL применяется. ACL применяется везде, например:

- На интерфейсе: *пакетная фильтрация*
- На линии Telnet: *ограничения доступа к маршрутизатору*
- VPN: *какой трафик нужно шифровать*
- QoS: *какой трафик обрабатывать приоритетнее*
- NAT: *какие адреса транслировать*

Для применения ACL для всех этих компонентов нужно понять как они работают. И мы в первую очередь будем касаться пакетной фильтрации. Применительно к пакетной фильтрации, ACL размещаются на интерфейсах, сами они создаются независимо, а уже потом они прикручиваются к интерфейсу. Как только вы его прикрутили к интерфейсу маршрутизатор начинает просматривать трафик. Маршрутизатор рассматривает трафик как входящий и исходящий. Тот трафик, который входит в маршрутизатор называется входящим, тот который из него выходит — исходящий. Соответственно ACL размещаются на входящем или на исходящем направлении.



Из вашей частной сети приходит пакет на интерфейс маршрутизатора fa0/1, маршрутизатор проверяет есть ли ACL на интерфейсе или нет, если он есть, то дальше обработка ведется по правилам списка доступа **строго в том порядке, в котором записаны выражения**, если список доступа разрешает проходить пакету, то в данном случае маршрутизатор отправляет пакет провайдеру через интерфейс fa0/0, если список доступа не разрешает проходить пакету, пакет уничтожается. Если списка доступа нет — пакет пролетает без всяких ограничений. Перед тем как отправить пакет провайдеру, маршрутизатор ещё проверяет интерфейс fa0/0 на наличие исходящего ACL. Дело в том, что ACL может быть прикреплен на интерфейсе как входящий или исходящий. К примеру у нас есть ACL с правилом запретить всем узлам в Интернете посылать в нашу сеть пакеты.
Так на какой интерфейс прикрепить данную ACL? Если мы прикрепим ACL на интерфейс fa0/1 как исходящий, это будет не совсем верно, хотя и ACL работать будет. На маршрутизатор приходит эхо-запрос для какого-то узла в частной сети, он проверяет на интерфейсе fa0/0 есть ли ACL, его нет, дальше проверяет интерфейс fa0/1, на данном интерфейсе есть ACL, он настроен как исходящий, всё верно пакет не проникает в сеть, а уничтожается маршрутизатором. Но если мы прикрепим ACL за интерфейсом fa0/0 как входящий, то пакет будет уничтожатся сразу как пришел на маршрутизатор. Последнее решение является правильным, так как маршрутизатор меньше нагружает свои вычислительные ресурсы. **Расширенные ACL нужно размещать как можно ближе к источнику, стандартные же как можно ближе к получателю**. Это нужно для того, чтобы не гонять пакеты по всей сети зря.

Сам же ACL представляет собой набор текстовых выражений, в которых написано **permit** (разрешить) либо**deny** (запретить), и обработка ведется строго в том порядке в котором заданы выражения. Соответственно когда пакет попадает на интерфейс он проверяется на первое условие, если первое условие совпадает с пакетом, дальнейшая его обработка прекращается. Пакет либо перейдет дальше, либо уничтожится.
Ещё раз, **если пакет совпал с условием, дальше он не обрабатывается**. Если первое условие не совпало, идет обработка второго условия, если оно совпало, обработка прекращается, если нет, идет обработка третьего условия и так дальше пока не проверятся все условия, **если никакое из условий не совпадает, пакет просто уничтожается**. Помните, в каждом конце списка стоит неявный deny any (запретить весь трафик). Будьте очень внимательны с этими правилами, которые я выделил, потому что очень часто происходят ошибки при конфигурации.

ACL разделяются на два типа:

- Стандартные (Standard): *могут проверять только адреса источников*
- Расширенные (Extended)**:** *могут проверять адреса источников, а также адреса получателей, в случае IP ещё тип протокола и TCP/UDP порты*

Обозначаются списки доступа либо номерами, либо символьными именами. ACL также используются для разных сетевых протоколов. Мы в свою очередь будем работать с IP. Обозначаются они следующим образом, нумерованные списки доступа:

- Стандартные: *от 1 до 99*
- Расширенные: *от 100 до 199*

Символьные ACL разделяются тоже на стандартные и расширенные. Расширенные напомню

могут проверять гораздо больше, нежели стандартные, но и работают они медленнее, так как придется заглядывать внутрь пакета, в отличии от стандартных где мы смотрим только поле Source Address (Адрес отправителя). При создании ACL каждая запись списка доступа обозначается порядковым номером, по умолчанию в рамках десяти (10, 20, 30 и т.д). Благодаря чему, можно удалить конкретную запись и на её место вставить другую, но эта возможность появилась в Cisco IOS 12.3, до 12.3 приходилось ACL удалять, а потом создать заново полностью.**Нельзя разместить более 1 списка доступа на интерфейс, на протокол, на направление**. Объясняю: если у нас есть маршрутизатор и у него есть интерфейс, мы можем на входящее направление для IP-протокола разместить только один список доступа, например под номером 10. Ещё одно правило, касающееся самих маршрутизаторов, **ACL не действует на трафик, сгенерированный самим маршрутизатором.**
Для фильтрации адресов в ACL используется WildCard-маска. Это обратная маска. Берем шаблонное выражение: 255.255.255.255 и отнимаем от шаблона обычную маску. 255.255.255.255-255.255.255.0, у нас получается маска 0.0.0.255, что является обычной маски 255.255.255.0, только 0.0.0.255 является WildCard маской.

- **Виды ACL**

- ***Динамический (Dynamic ACL)***

Позволяет сделать следующее, например у вас есть маршрутизатор, который подключен к какому-то серверу и нам нужно закрыть доступ к нему из внешнего мира, но в тоже время есть несколько человек, которые могут подключаться к серверу.
Мы настраиваем динамический список доступа, прикрепляем его на входящем направлении, а дальше людям, которым нужно подключиться, подключаться через Telnet к данному устройству, в результате динамический ACL открывает проход к серверу, и уже человек может зайти скажем через HTTP попасть на сервер. По умолчанию через 10 минут этот проход закрывается и пользователь вынужден ещё раз выполнить Telnet чтобы подключиться к устройству.

- ***Рефлексивный (Reflexive ACL)***

Здесь ситуация немножко отличается, когда узел в локальной сети отправляет TCP запрос в Интернет, у нас должен быть открытый проход, чтобы пришел TCP ответ для установки соединения. Если прохода не будет — мы не сможем установить соединение, и вот этим проходом могут воспользоваться злоумышленники, например проникнуть в сеть. Рефлексивные ACL работают таким образом, блокируется полностью доступ (deny any) но формируется ещё один специальный ACL, который может читать параметры пользовательских сессий, которые сгенерированны из локальной сети и для них открывать проход в deny any, в результате получается что из Интернета не смогут установить соединение. А на сессии сгенерированны из локальной сети будут приходить ответы.

- ***Ограничение по времени (Time-based ACL)***

Обычный ACL, но с ограничением по времени, вы можете ввести специальное расписание, которое активирует ту или иную запись списка доступа. И сделать такой фокус, например пишем список доступа, в котором запрещаем HTTP-доступ в течении рабочего дня и вешаем его на интерфейс маршрутизатора, то есть, сотрудники предприятия пришли на работу, им закрывается HTTP-доступ, рабочий день закончился, HTTP-доступ открывается,
пожалуйста, если хотите — сидите в Интернете.

- *Настройка*

Сами ACL создаются отдельно, то есть это просто некий список, который создается в глобальном конфиге, потом он присваивается к интерфейсу и только тогда он и начинает работать.
Необходимо помнить некоторые моменты, для того, чтобы правильно настроить списки доступа:

- Обработка ведется строго в том порядке, в котором записаны условия
- Если пакет совпал с условием, дальше он не обрабатывается
- В конце каждого списка доступа стоит неявный deny any (запретить всё)
- Расширенные ACL нужно размещать как можно ближе к источнику, стандартные же как можно ближе к получателю
- Нельзя разместить более 1 списка доступа на интерфейс, на протокол, на направление
- ACL не действует на трафик, сгенерированный самим маршрутизатором
- Для фильтрации адресов используется WildCard маска

- ***Стандартный список доступа***

Router(config)#**access-list** <номер списка от 1 до 99> **{permit | deny | remark} {address | any | host}**[source-wildcard] **[log]**

- permit: *разрешить*
- deny: *запретить*

- remark: *комментарий о списке доступа*
- address: *запрещаем или разрешаем сеть*
- any: *разрешаем или запрещаем всё*
- host: *разрешаем или запрещаем хосту*
- source-wildcard: *WildCard маска сети*
- log: *включаем логгирование пакеты проходящие через данную запись ACL*

- ***Расширенный список доступа***

Router(config)#**access-list** <номер списка от 100 до 199> **{permit | deny | remark}** protocol source [source-wildcard] [**operator** operand] [**port** <порт или название протокола> **[established]**

- protocol source: *какой протокол будем разрешать или закрывать (ICMP, TCP, UDP, IP, OSPF и т.д)*
- deny: *запретить*
- operator:
  A.B.C.D — адрес получателя
  any — любой конечный хост
  eq — только пакеты на этом порте
  gt — только пакеты с большим номером порта
  host — единственный конечный хост
  lt — только пакеты с более низким номером порта
  neq — только пакеты не на данном номере порта
  range — диапазон портов
- port: *номер порта (TCP или UDP), можно указать имя*
- established: *разрешаем прохождение TCP-сегментов, которые являются частью уже созданной TCP-сессии*

- ***Прикрепляем к интерфейсу***

Router(config-if)#**ip access-group** <номер списка или имя ACL> **{in | out}**

- in: *входящее направление*

- out: *исходящее направление*

- ***Именованные списки доступа***

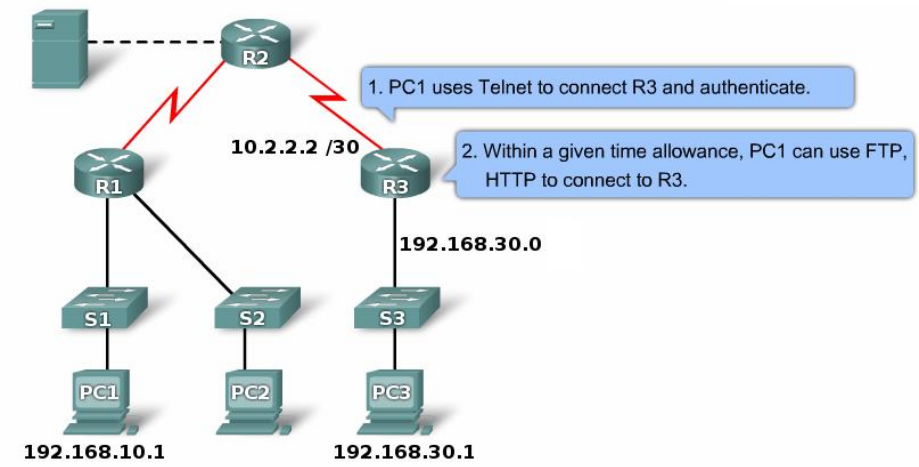
Router(config)#**ip access-list {standard | extended} {<номер ACL> | <имя ACL>}**
Router(config-ext-nacl)# **{default | deny | exit | no | permit | remark}**

- standard: *стандартный ACL*
- extended: *расширенный ACL*
- default: *установить команду в значение по умолчанию*

- ***Ограничение доступа к маршрутизатору***

R(config)#**line vty** 0 4 — переходим в режим настройки виртуальных линий.
R(config-line)#**password** <пароль>
R(config-line)#**login**
R(config-line)#**access-class** 21 **in** — настраиваем логин и пароль, а также закрепляем список доступа с разрешенными IP-адресами.

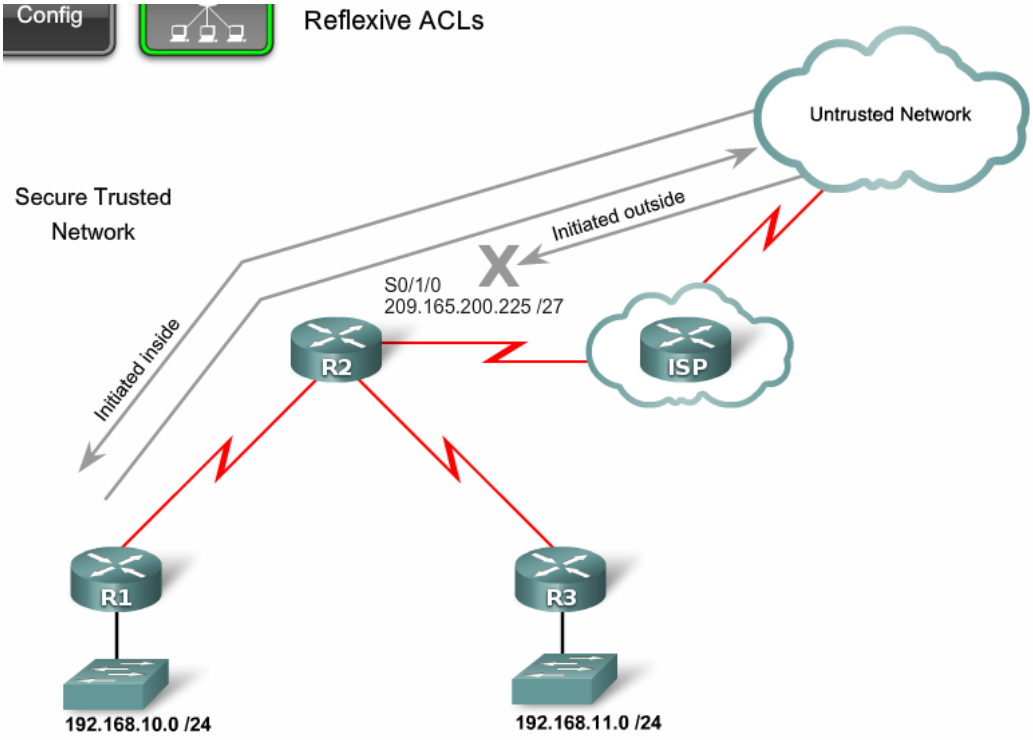
- ***Динамические списки доступа***



R3(config)#**username** Student **password 0** cisco — создаем пользователей для подключения через Telnet.
R3(config)#**access-list** 101 **permit tcp any host** 10.2.2.2 **eq telnet**

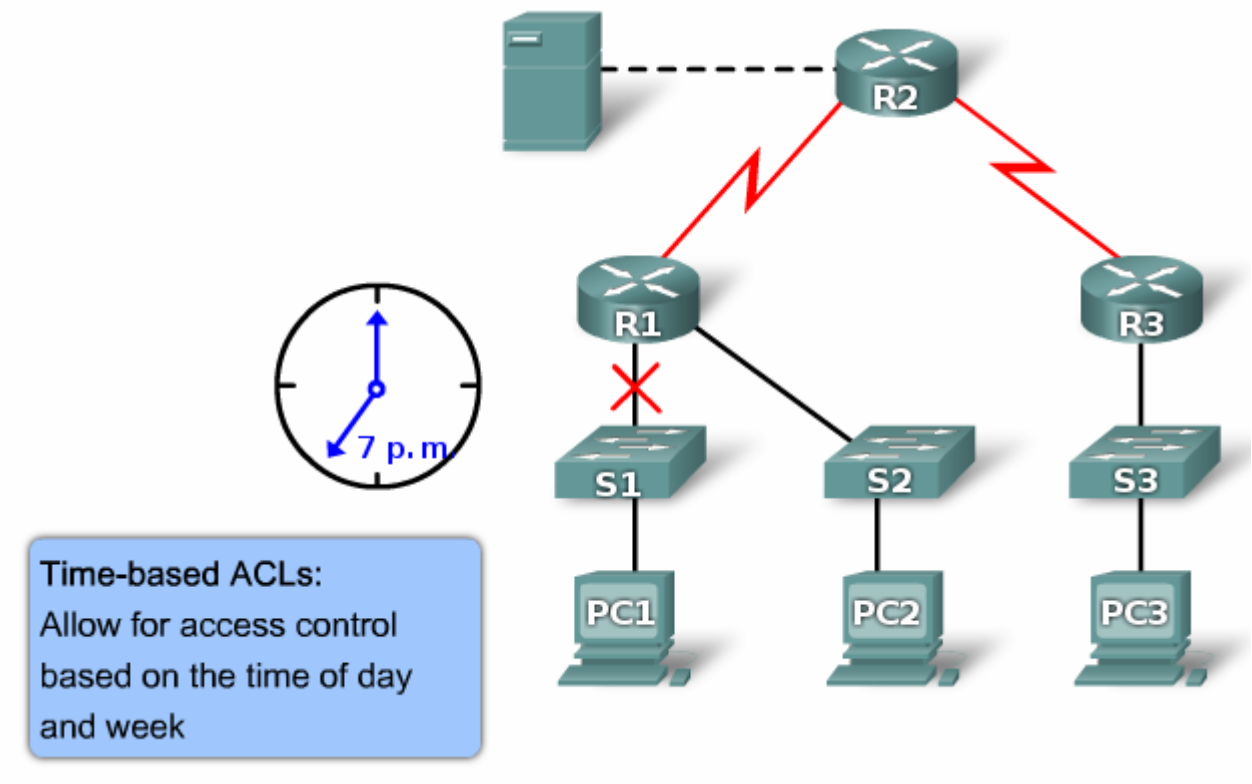
R3(config)#**access-list** 101 **dynamic testlist timeout 15 permit ip** 192.168.10.0 0.0.0.255 192.168.30.0 0.0.0.255 — разрешаем подключаться к серверу по Telnet всем узлам.
R3(config)#interface serial 0/0/1
R3(config-if)#**ip access-group** 101 in — закрепляем 101 ACL за интерфейсом в входящем направлении.
R3(config)#**line vty** 0 4
R3(config-line)#**login local**
R3(config-line)#**autocommand access-enable host timeout** 5 — как только пользователь аутентифицируется, сеть 192.168.30.0 будет доступна, через 5 минут бездействия сеанс закроется.

- ***Рефлексивные списки доступа***



R2(config)#**ip access-list extended** OUTBOUNDFILTERS
R2(config-ext-nacl)#**permit tcp** 192.168.0.0 0.0.255.255 **any reflect** TCPTRAFFIC
R2(config-ext-nacl)#**permit icmp** 192.168.0.0 0.0.255.255 **any reflect** ICMPTRAFFIC — заставляем маршрутизатор отслеживать трафик, который инициировался изнутри.
R2(config)#**ip access-list extended** INBOUNDFILTERS
R2(config-ext-nacl)#**evaluate** TCPTRAFFIC
R2(config-ext-nacl)#**evaluate** ICMPTRAFFIC — создаем входящую политику, которая требует, чтобы маршрутизатор проверял входящий трафик, чтобы видеть инициировался ли изнутри и связываем TCPTRAFFIC к INBOUNDFILTERS.
R2(config)#**interface** serial 0/1/0
R2(config-if)#**ip access-group** INBOUNDFILTERS in
R2(config-if)#**ip access-group** OUTBOUNDFILTERS out — применяем входящий и исходящий ACL на интерфейс.

- ***Ограничение по времени***



R1(config)#**time-range** EVERYOTHERDAY
R1(config-time-range)#**periodic** Monday Wednesday Friday 8:00 to 17:00 — создаем список времени, в котором добавляем дни недели и время.
R1(config)#**access-list** 101 **permit tcp** 192.168.10.0 0.0.0.255 **any eq telnet time-range** EVERYOTHERDAY — применяем time-range к ACL.
R1(config)#interface s0/0/0
R1(config-if)#**ip access-group** 101 out — закрепляем ACL за интерфейсом.

- ***Поиск проблем***

R#**show access-lists** {ACL номер | имя} — смотрим информацию о списке доступа.
R#**show access-lists** — смотрим все списки доступа на маршрутизаторе.

- ***Пример***

Router#**show access-lists**

Extended IP access list **nick**
permit ip host 172.168.1.1 host 10.0.0.5
deny ip any any (16 match(es))
Standard IP access list **nick5**
permit 172.16.0.0 0.0.255.255

Мы видим что у нас есть два ACL (стандартный и расширенный) под названиями nick и nick5. Первый список разрешает хосту 172.16.1.1 обращаться по IP (это значит что разрешены все протоколы работающие поверх IP) к хосту 10.0.0.5. Весь остальной трафик запрещен показывает команда deny ip any any. Рядом с этим условием в нашем примере пишет (16 match(es)). Это показывает что 16 пакетов попали под это условие.
Второй ACL разрешает проходить трафик от любого источника в сети 172.16.0.0/16.

2. Порядок выполнения работы:

1. В программе CiscoPacketTracer создайте сеть со следующими характеристиками

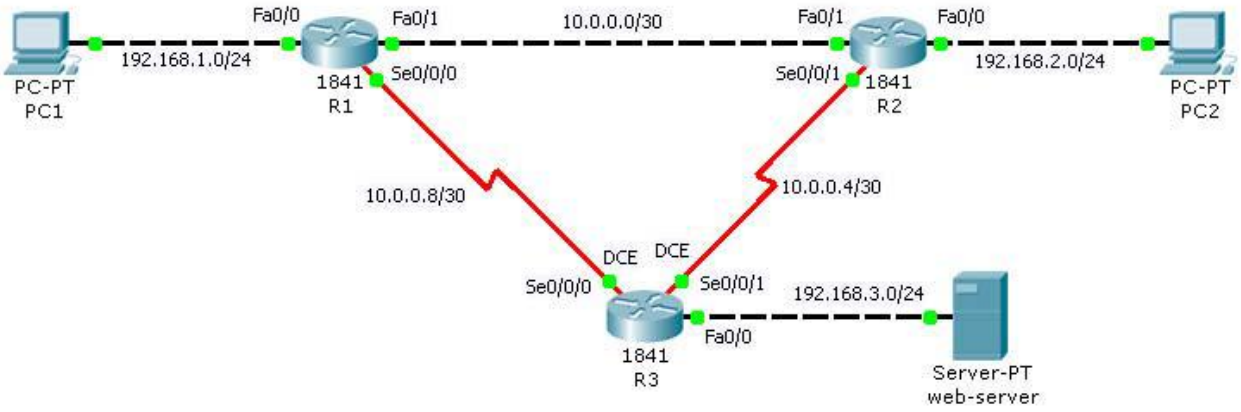


Таблица сетевых адресов.

Device	Interface	IP Address	Mask	Default Gateway
R1	Fa0/0	192.168.1.1	255.255.255.0	N/A
	Fa0/1	10.0.0.1	255.255.255.252	N/A
	Se0/0/0	10.0.0.9	255.255.255.252	N/A
R2	Fa0/0	192.168.2.1	255.255.255.0	N/A
	Fa0/1	10.0.0.2	255.255.255.252	N/A
	Se0/0/1	10.0.0.6	255.255.255.252	N/A
R3	Fa0/0	192.168.3.1	255.255.255.0	N/A
	Se0/0/0	10.0.0.10	255.255.255.252	N/A
	Se0/0/1	10.0.0.5	255.255.255.252	N/A
PC1	N/A	192.168.1.10	255.255.255.0	192.168.1.1
PC2	N/A	192.168.2.10	255.255.255.0	192.168.2.1
Web-server	N/A	192.168.3.10	255.255.255.0	192.168.3.1

2. Настройте на маршрутизаторе R1 стандартный ACL, запрещающий устройству PC1 взаимодействовать с устройствами из других сетей.

2.1. Зайдите в режим глобальной конфигурации маршрутизатора.

```
R1>enable
R1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#
```

2.2. Создайте стандартный ACL.

```
R1(config)#access-list 1 deny 192.168.1.10 0.0.0.0
```

access-list	Команда создания ACL
1	Номер ACL
deny	Команда «запретить»
192.168.1.10	Адрес, к которому надо применить команду
0.0.0.0	Wildcard маска

```
R1(config)#access-list 1 permit any
```

2.3. Установите ACL на интерфейсе fa0/0 маршрутизатора R1.

```
R1(config)#interface fa 0/0
R1(config-if)#ip access-group 1 in
```

3. Проверьте правильность настройки стандартного ACL.

3.1. Зайдите в эмулятор командной строки на устройстве PC1.

3.2. С помощью утилиты ping проверьте возможность взаимодействия устройства PC1 с любым конечным устройством сети.

Если PC1 не получает эхо ответы от другого устройства, ACL настроен правильно.

4. Настройте на маршрутизаторе R3 расширенный ACL, запрещающий устройству PC2 обращаться к веб-серверу по протоколу HTTP.

4.1. Зайдите в режим глобальной конфигурации маршрутизатора.

```
R3>enable
R3#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R3(config)#
```

4.2. Создайте стандартный ACL.

```
R3(config)#access-list 101 deny tcp 192.168.2.10 0.0.0.0 192.168.3.10
0.0.0.0 eq www
```

где, *access-list* *Команда создания ACL*
101 *Номер ACL*
deny *Команда «запретить»*
tcp *Протокол транспортного уровня*
192.168.2.10 *Адрес источника*

0.0.0.0 Wildcard маска для адреса источника

192.168.3.10 Адрес получателя

0.0.0.0 Wildcard маска для адреса получателя

eq www Порт назначения, по которому нужно запретить взаимодействие

```
R3(config)#access-list 101 permit ip any any
R3(config)#access-list 101 permit icmp any any
```

2.3. Установите ACL на интерфейсе s0/0/1 маршрутизатора R1.

```
R3(config)#interface serial 0/0/1
```

```
R3(config-if)#ip access-group 101 in
```
5. Проверьте правильность настройки расширенного ACL.
5.1. Зайдите в эмулятор командной строки на устройстве PC2.
3.2. С помощью утилиты ping проверьте возможность взаимодействия устройства PC1 с любым конечным устройством сети.
3.3. С помощью эмулятора браузера попробуйте загрузить сайт **www.site.ru**
Если устройство PC2 получает эхо-ответы от сервера, но страницу загрузить не удаётся, значит ACL настроен правильно.
6. Сохраните конфигурацию устройств.

```
Router#copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
Router#
```

3. Контрольные вопросы

1. Что такое ACL?
2. Какой адрес является критерием для разрешения/запрещения пакета?
3. Где применяются ACL?
4. Как задать элемент ACL и что такое инверсная маска?
5. Что фильтруют расширенные ACL?
6. Какую дополнительную функциональность имеют расширенные ACL по сравнению со стандартными?
7. Можно ли, используя расширенные ACL, наложить ограничения на трафик к определённой TCP/IP службе?
8. Опишите процедуру создания именованного ACL.

4. Содержание отчета

1) титульный лист;

2) формулировку цели работы;

3) описание результатов выполнения;

4) выводы, согласованные с целью работы.