Лабораторная работа №4

Тема: Настройка DNS, HTTP, MAIL

Теоретические сведения:

TCP

Transmission Control Protocol (TCP) (протокол управления передачей) — один из основных сетевых протоколов Интернета, предназначенный для управления передачей данных в сетях и подсетях TCP/IP.

Выполняет функции протокола транспортного уровня модели OSI.

TCP — это транспортный механизм, предоставляющий поток данных, с предварительной установкой соединения, за счёт этого дающий уверенность в достоверности получаемых данных, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета. В отличие от UDP гарантирует целостность передаваемых данных и уведомление отправителя о результатах передачи.

Реализация TCP, как правило, встроена в ядро ОС, хотя есть и реализации TCP в контексте приложения.

Когда осуществляется передача от компьютера к компьютеру через Интернет, TCP работает на верхнем уровне между двумя конечными системами, например, браузером и веб-сервером. Также TCP осуществляет надежную передачу потока байтов от одной программы на некотором компьютере к другой программе на другом компьютере. Программы для работы с электронной почтой и обмена файлами также используют TCP. TCP контролирует длину сообщения, скорость обмена сообщениями, сетевой трафик.

Схема заголовка сегмента:

Бит	**0 — 3**	**4 — 9**	**10 — 15**	**16 — 31**
0	Порт источника			Порт назначения
32	Номер последовательности			
64	Номер подтверждения			
96	Смещение данных	Зарезервировано	Флаги	Размер Окна

128	Контрольная сумма	Указатель важности
160	Опции (необязательное, но используется практически всегда)	
160/192+	Данные	

Механизм действия протокола

В отличие от традиционной альтернативы — UDP, который может сразу же начать передачу пакетов, TCP устанавливает соединения, которые должны быть созданы перед передачей данных. TCP соединение можно разделить на 3 стадии:

- Установка соединения
- Передача данных
- Завершение соединения

Установка соединения

Процесс начала сеанса TCP — обозначаемое как «рукопожатие» (handshake), состоит из 3 шагов.

3) Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN.
   - Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет для обслуживания нового клиента.
   - В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED. Иначе сервер посылает клиенту сегмент с флагом RST.
4) Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK.
   1. Если он одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED. Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться.

2. В случае, когда клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

5) Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED.
   - В противном случае после тайм-аута он закрывает сокет и переходит в состояние CLOSED.

Процесс называется «трехэтапным согласованием» («three way handshake»).

Передача данных

При обмене данными приемник использует номер последовательности, содержащийся в получаемых сегментах, для восстановления их исходного порядка. Приемник уведомляет передающую сторону о номере последовательности, до которой он успешно получил данные, включая его в поле «номер подтверждения». Все получаемые данные, относящиеся к промежутку подтвержденных последовательностей, игнорируются. Если полученный сегмент содержит номер последовательности больший, чем ожидаемый, то данные из сегмента буферизируются, но номер подтвержденной последовательности не изменяется.

Завершение соединения

Завершение соединения можно рассмотреть в три этапа:

1. Посылка серверу от клиента флага FIN на завершение соединения.
2. Сервер посылает клиенту флаги ответа ACK , FIN, что соединение закрыто.
3. После получения этих флагов клиент закрывает соединение и в подтверждение отправляет серверу ACK , что соединение закрыто.

DNS

DNS (англ. Domain Name System — система доменных имён) — компьютерная распределённая система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста (компьютера

или устройства), получения информации о маршрутизации почты, обслуживающих узлах для протоколов в домене.

Распределённая база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих по определённому протоколу.

Основой DNS является представление об иерархической структуре доменного имени и зонах. Каждый сервер, отвечающий за имя, может делегировать ответственность за дальнейшую часть домена другому серверу (с административной точки зрения — другой организации или человеку), что позволяет возложить ответственность за актуальность информации на серверы различных организаций (людей), отвечающих только з*а «свою»* часть доменного имени.

Начиная с 2010 года, в систему DNS внедряются средства проверки целостности передаваемых данных, называемые DNS Security Extensions (DNSSEC). Передаваемые данные не шифруются, но их достоверность проверяется криптографическими способами. Внедряемый стандарт DANE обеспечивает передачу средствами DNS достоверной криптографической информации (сертификатов), используемых для установления безопасных и защищённых соединений транспортного и прикладного уровней.

DNS обладает следующими характеристиками:

Распределённость администрирования. Ответственность за разные части иерархической структуры несут разные люди или организации.

Распределённость хранения информации. Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в его зону ответственности и (возможно) адреса корневых DNS-серверов.

Кеширование информации. Узел может хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть.

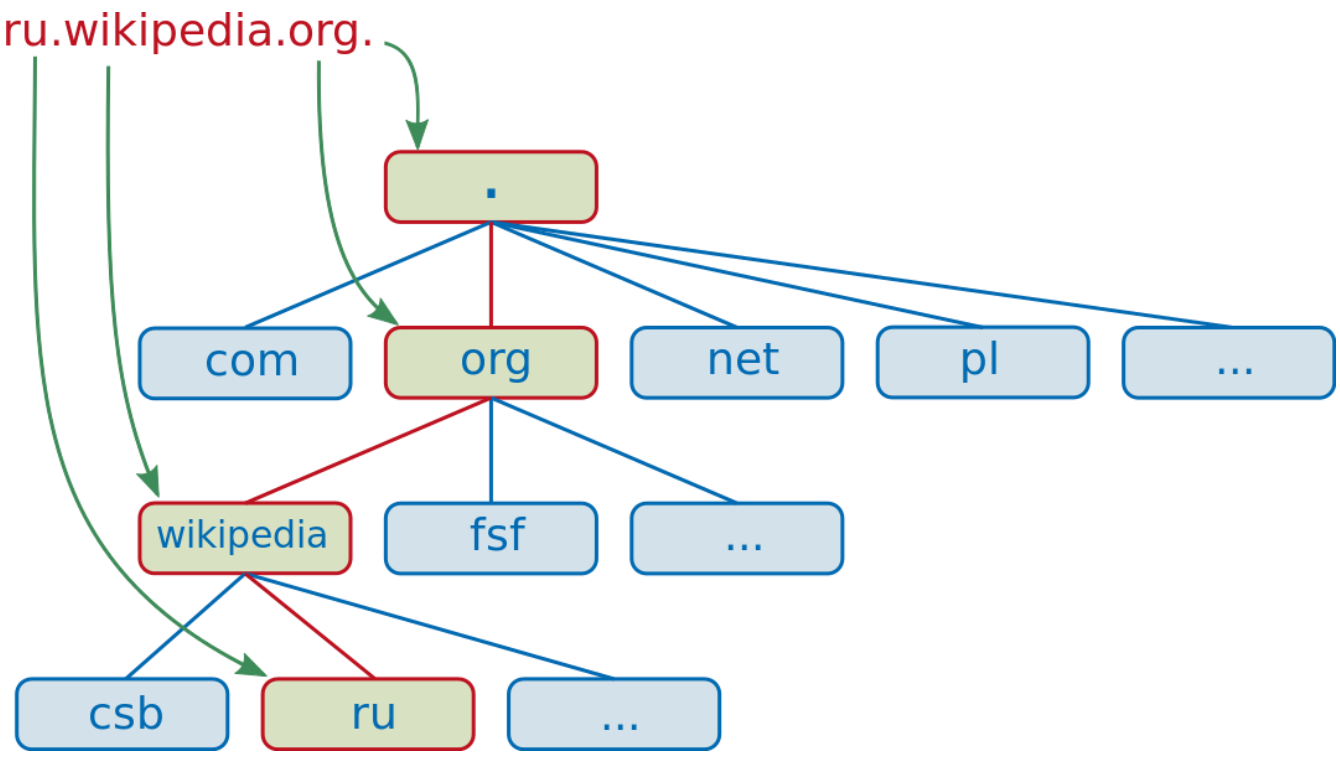
Иерархическая структура, в которой все узлы объединены в дерево, и каждый узел может или самостоятельно определять работу нижестоящих узлов,

или делегировать (передавать) их другим узлам.

Резервирование. За хранение и обслуживание своих узлов (зон) отвечают (обычно) несколько серверов, разделённые как физически, так и логически, что обеспечивает сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

DNS важна для работы Интернета, так как для соединения с узлом необходима информация о его IP-адресе, а для людей проще запоминать буквенные (обычно осмысленные) адреса, чем последовательность цифр IP-адреса. В некоторых случаях это позволяет использовать виртуальные серверы, например, HTTP-серверы, различая их по имени запроса. Первоначально преобразование между доменными и IP-адресами производилось с использованием специального текстового файла hosts, который составлялся централизованно и автоматически рассылался на каждую из машин в своей локальной сети. С ростом Сети возникла необходимость в эффективном, автоматизированном механизме, которым и стала DNS.

DNS была разработана Полом Мокапетрисом в 1983 году(рис 1);



Система DNS содержит иерархию DNS-серверов, соответствующую иерархии зон. Каждая зона поддерживается как минимум одним авторитетным сервером DNS (от англ. authoritative — авторитетный), на котором расположена информация о домене.

Имя и IP-адрес не тождественны — один IP-адрес может иметь множество имён, что позволяет поддерживать на одном компьютере множество веб-сайтов (это называется виртуальный хостинг). Обратное тоже справедливо — одному имени может быть сопоставлено множество IP-адресов: это позволяет создавать балансировку нагрузки.

Для повышения устойчивости системы используется множество серверов, содержащих идентичную информацию, а в протоколе есть средства, позволяющие поддерживать синхронность информации, расположенной на разных серверах. Существует 13 корневых серверов, их адреса практически не изменяются.

Протокол DNS использует для работы TCP- или UDP-порт 53 для ответов на запросы.

Ключевыми понятиями DNS являются:

• **Домен** (англ. domain — область)— узел в дереве имён, вместе со всеми подчинёнными ему узлами (если таковые имеются), то есть именованная ветвь или поддерево в дереве имен. Структура доменного имени отражает порядок следования узлов в иерархии; доменное имя читается справа налево от младших доменов к доменам высшего уровня (в порядке повышения значимости): вверху находится корневой домен, ниже идут домены первого уровня, затем — домены второго уровня, третьего и т. д. (например, для адреса ru.wikipedia.org. домен первого уровня — org, второго wikipedia, третьего ru).

• **Поддомен**(англ. subdomain) — подчинённый домен (например, wikipedia.org — поддомен доменаorg, а ru.wikipedia.org — домена wikipedia.org). Теоретически такое деление может достигать глубины 127 уровней, а каждая метка может содержать до 63 символов, пока общая длина вместе с точками не достигнет 254 символов. Но на практике регистраторы доменных имён используют более строгие ограничения.

• **Ресурсная запись** — единица хранения и передачи информации в DNS.

• **Зона** — часть дерева доменных имен (включая ресурсные записи), размещаемая как единое целое на некотором сервере доменных имен, а чаще — одновременно на нескольких серверах.

• **Делегирование** — операция передачи ответственности за часть дерева доменных имен другому лицу или организации. За счет делегирования в DNS обеспечивается распределенность администрирования и хранения.

• **DNS-сервер** — специализированное ПО для обслуживания DNS, а также компьютер, на котором это ПО выполняется. DNS-сервер может быть ответственным за некоторые зоны и/или может перенаправлять запросы вышестоящим серверам.

• **DNS-клиент** — специализированная библиотека (или программа) для работы с DNS. В ряде случаев DNS-сервер выступает в роли DNS-клиента.

• **Авторитетность** (англ. authoritative) — признак размещения зоны на DNS-сервере. Ответы DNS-сервера могут быть двух типов: авторитетные (когда сервер заявляет, что сам отвечает за зону) инеавторитетные (англ. non-authoritative), когда сервер обрабатывает запрос, и возвращает ответ других серверов. В некоторых случаях вместо передачи запроса дальше DNS-сервер может вернуть уже известное ему (по запросам ранее) значение (режим кеширования).

• **DNS-запрос** (англ. DNS query) — запрос от клиента (или сервера) серверу. Запрос может быть рекурсивным (требующим полного поиска) или нерекурсивным (или итеративным) — не требующим полного поиска.

HTTP

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом. Порт: 80/8080.

HTTP в настоящее время повсеместно используется во Всемирной паутине для получения информации с веб-сайтов. HTTP используется также в качестве «транспорта» для других протоколов прикладного уровня, таких как SOAP, XML-RPC, WebDAV.

Основным объектом манипуляции в HTTP является ресурс, на который указывает URI (англ. Uniform Resource Identifier) в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы, но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. (В частности для этого используется HTTP-заголовок.) Именно благодаря

возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

HTTP — протокол прикладного уровня, аналогичными ему являются FTP и SMTP. Обмен сообщениями идёт по обыкновенной схеме «запрос-ответ». Для идентификации ресурсов HTTP использует глобальные URI. В отличие от многих других протоколов, HTTP не сохраняет своего состояния. Это означает отсутствие сохранения промежуточного состояния между парами «запрос-ответ». Компоненты, использующие HTTP, могут самостоятельно осуществлять сохранение информации о состоянии, связанной с последними запросами и ответами (например, «куки» на стороне клиента, «сессии» на стороне сервера). Браузер, посылающий запросы, может отслеживать задержки ответов. Сервер может хранить IP-адреса и заголовки запросов последних клиентов. Однако сам протокол не осведомлён о предыдущих запросах и ответах, в нём не предусмотрена внутренняя поддержка состояния, к нему не предъявляются такие требования.

Всё программное обеспечение для работы с протоколом HTTP разделяется на три большие категории:

Серверы как основные поставщики услуг хранения и обработки информации (обработка запросов).

Клиенты — конечные потребители услуг сервера (отправка запроса).

Прокси для выполнения транспортных служб.

Каждое HTTP-сообщение состоит из трёх частей, которые передаются в указанном порядке:

Стартовая строка (англ. Starting line) — определяет тип сообщения;

Заголовки (англ. Headers) — характеризуют тело сообщения, параметры передачи и прочие сведения;

Тело сообщения (англ. Message Body) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой.

Заголовки и тело сообщения могут отсутствовать, но стартовая строка является обязательным элементом, так как указывает на тип запроса/ответа.

Методы

Метод HTTP (англ. HTTP Method) — последовательность из любых символов, кроме управляющих и разделителей, указывающая на основную операцию над ресурсом. Обычно метод представляет собой короткое английское слово, записанное заглавными буквами. Обратите внимание, что название метода чувствительно к регистру.

Каждый сервер обязан поддерживать как минимум методы GET и HEAD. Если сервер не распознал указанный клиентом метод, то он должен вернуть статус 501 (Not Implemented). Если серверу метод известен, но он неприменим к конкретному ресурсу, то возвращается сообщение с кодом 405 (Method Not Allowed). В обоих случаях серверу следует включить в сообщение ответа заголовок Allow со списком поддерживаемых методов.

Кроме методов GET и HEAD, часто применяется метод POST.

MAIL

Электро́нная по́чта (англ. email, e-mail, от англ. electronic mail) — технология и предоставляемые ею услуги по пересылке и получению электронных сообщений (называемых «письма» или «электронные письма») по распределённой (в том числе глобальной) компьютерной сети.

Электронная почта по составу элементов и принципу работы практически повторяет систему обычной (бумажной) почты, заимствуя как термины (почта, письмо, вложение, ящик, доставка и другие), так и характерные особенности — простоту использования, задержки передачи сообщений, достаточную надёжность и в то же время отсутствие гарантии доставки.

Достоинствами электронной почты являются: легко воспринимаемые и запоминаемые человеком адреса вида имя_пользователя@имя_домена (например somebody@example.com); возможность передачи как простого текста, так и

форматированного, а также произвольных файлов; независимость серверов (в общем случае они обращаются друг к другу непосредственно); достаточно высокая надёжность доставки сообщения; простота использования человеком и программами.

Недостатки электронной почты: наличие такого явления, как спам (массовые рекламные и вирусные рассылки); теоретическая невозможность гарантированной доставки конкретного письма; возможные задержки доставки сообщения (до нескольких суток); ограничения на размер одного сообщения и на общий размер сообщений в почтовом ящике (персональные для пользователей).

В настоящее время любой начинающий пользователь может завести свой бесплатный электронный почтовый ящик, достаточно зарегистрироваться на одном из интернет-порталов.

SMTP

Общепринятым в мире протоколом обмена электронной почтой является SMTP (англ. Simple mail transfer protocol — простой протокол передачи почты).

SMTP впервые был описан в RFC 821 (1982 год); последнее обновление в RFC 5321 (2008) включает масштабируемое расширение — ESMTP (англ. Extended SMTP). В настоящее время под «протоколом SMTP», как правило, подразумевают и его расширения. Протокол SMTP предназначен для передачи исходящей почты с использованием порта TCP 25.

В то время, как электронные почтовые серверы и другие агенты пересылки сообщений используют SMTP для отправки и получения почтовых сообщений, работающие на пользовательском уровне клиентские почтовые приложения обычно используют SMTP только для отправки сообщений на почтовый сервер для ретрансляции. Для получения сообщений клиентские приложения обычно используют либо POP (англ. Post Office Protocol — протокол почтового отделения), либо IMAP (англ. Internet Message Access Protocol), либо патентованные системы (такие как Microsoft Exchange и Lotus Notes/Domino) для доступа к учетной записи

своего почтового ящика на сервере.

Протоколы получения почты

После попадания почты на конечный сервер, он осуществляет временное или постоянное хранение принятой почты. Существует две различные модели работы с почтой: концепция почтового хранилища (ящика) и почтового терминала.

POP и IMAP — наиболее распространенные Интернет-протоколы для извлечения почты. Практически все современные клиенты и сервера электронной почты поддерживают оба стандарта.

POP3

POP3 (англ. Post Office Protocol Version 3 — протокол почтового отделения, версия 3) — стандартный Интернет-протокол прикладного уровня, используемый клиентами электронной почты для извлечения электронного сообщения с удаленного сервера по TCP/IP-соединению.

POP3-сервер прослушивает общеизвестный порт 110.

В концепции почтового хранилища почта на сервере хранится временно, в ограниченном объёме (аналогично почтовому ящику для бумажной почты), а пользователь периодически обращается к ящику и «забирает» письма (то есть почтовый клиент скачивает копию письма к себе и удаляет оригинал из почтового ящика). На основании этой концепции действует протокол POP3.

IMAP

IMAP (англ. Internet Message Access Protocol) — протокол прикладного уровня для доступа к электронной почте.

Базируется на транспортном протоколе TCP и использует порт 143.

Концепция почтового терминала подразумевает, что вся корреспонденция, связанная с почтовым ящиком (включая копии отправленных писем), хранится на сервере, а пользователь обращается к хранилищу (иногда его по традиции также называют «почтовым ящиком») для просмотра корреспонденции (как новой, так и архива) и написания новых писем (включая ответы на другие письма). На этом

принципе действует протокол IMAP и большинство веб-интерфейсов бесплатных почтовых служб. Подобное хранение почтовой переписки требует значительно бóльших мощностей от почтовых серверов, в результате, во многих случаях происходит разделение между почтовыми серверами, пересылающими почту, и серверами хранения писем.

Различия

Основываясь на работе протоколов можно разделить их по двум основным критериям:

Производительность сервера — в данном случае, IMAP более требователен к ресурсам нежели POP3, так как вся работа по обработке почты (такая как поиск) ложится на плечи сервера, POP3 только передает почту клиенту;

Пропускной способности канала — тут IMAP в выигрыше; POP3 передает тела всех писем целиком, тогда как IMAP — только заголовки писем, а остальное — по запросу.

В определённых условиях сервер хранения писем может быть настроен на поведение, подобное клиенту: такой сервер обращается к почтовому серверу по протоколу POP3 и забирает почту себе. Подобные решения используются обычно в малых организациях, в которых нет инфраструктуры для развёртывания полноценных почтовых серверов; в этом случае используется локальный сервер для хранения почты и почтовый сервер провайдера, предоставляющий услугу получения почты по POP3 (например, с помощью fetchmail). Основным недостатком подобного решения является задержка в доставке (так как забирающее почту ПО обращается на сервера с некоторой задержкой) — например, POP3 connector из Exchange 2003 Server в составе Windows SBS не позволяет через интерфейс конфигурирования выставить интервал менее 15 минут.

Практическое задание:

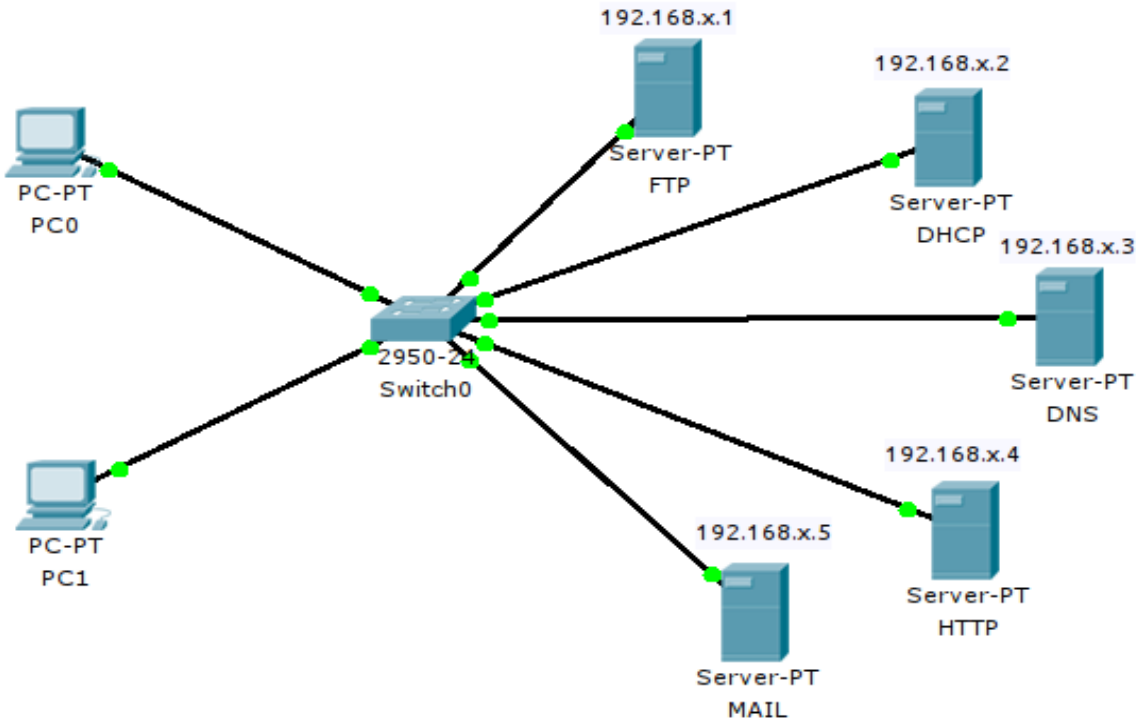


Рис. 1

Исходные данные:

x = <номер зач. кн.> mod 30

user = <фамилия латинницой>

password = *user*

group = <группа латинницой>

Ход работы:

Создаем топологию как показано на рис. 1. Настройки DHCP взять из лабораторной работы №3

Назначим IP адреса серверам следующим образом:

Сервер	**IP адрес**	**Маска (Subnet mask)**
FTP	192.168.*x*.1	255.255.255.0
DHCP	192.168.*x*.2	255.255.255.0
DNS	192.168.*x*.3	255.255.255.0
HTTP	192.168.*x*.4	255.255.255.0
MAIL	192.168.*x*.5	255.255.255.0

Настройка сервера DNS

Имя	**Тип записи**	**IP адрес**
ftp.<*user*>.kpi.ua	A	192.168.*x*.1
www.<*user*>.kpi.ua	A	192.168.*x*.4
smtp.<*group*>.kpi.ua	A	192.168.*x*.5
pop.<*group*>.kpi.ua	A	192.168.*x*.5

Настройка сервера DNS

Отредактировать index.html таким образом, чтобы отображались все уровни OSI модели в виде списка или таблицы.

Примечание. Оформление по желанию

Настройка сервера MAIL

1) Domain name: <*group*>.kpi.ua

2) Создать 2 пользователя

User	**Password**	**Примечание**
<*name*>	<*password*>	Ваш почтовый ящик
all	<*group*>	Ящик для всей группы

На PC0 устройстве сделать следующие настройки в приложении E-Mail.

Ваше имя	**ФИО**
Email адрес	<*name*>@<*group*>.kpi.ua
Входящий Mail сервер	pop.<*group*>.kpi.ua
Исходящий Mail сервер	smtp.<*group*>.kpi.ua
Имя пользователя	<*name*>
Пароль	<*password*>

Сделать аналогические настройки на устройстве PC1 только для почтового ящика all@<group>.kpi.ua

Проверка выполнения работы:

Получить IP настройки для устройств PC0, PC1

Проверить доступность всех серверов с помощью передачи PDU в режиме симуляции

Проверить работу DNS. В командной строке устройства PC0 ввести команду nslookup и в строки приглашения последовательно вводить доменные имена, которые были заданы при конфигурации DNS сервера

В Web браузере устройства PC0 просмотреть содержимое сайта www.<*name*>.kpi.ua. С помощью симуляции увидеть работу протоколов DNS, TCP, HTTP (тип пакетов, информацию о каждом пакете)

В приложении E-Mail устройства PC0 отправить письмо по адресу all@<group>.kpi.ua. Письмо должно информировать всю группу о следующей лекции предмета «Организация компьютерных сетей» (указать также дату, место и время).

С помощью устройства PC1 получить почту с почтового сервера и прочитать содержимое почтового ящика all@<*group*>.kpi.ua. Ответить отправителю с подтверждением о присутствии на лекции.

Список литературы:

1. Hughes L Internet e-mail Protocols, Standards and Implementation. — Artech House Publishers, 1998. — ISBN 0-89006-939-5
2. Hunt C sendmail Cookbook. — O'Reilly Media, 2003. — ISBN 0-596-00471-0
3. Johnson K Internet Email Protocols: A Developer's Guide. — Addison-Wesley Professional, 2000. — ISBN 0-201-43288-9
4. Loshin P Essential Email Standards: RFCs and Protocols Made Practical. — John Wiley & Sons, 1999. — ISBN 0-471-34597-0
5. Rhoton J Programmer's Guide to Internet Mail: SMTP, POP, IMAP, and LDAP. — Elsevier, 1999. — ISBN 1-55558-212-5
6. Wood D Programming Internet Mail. — O'Reilly, 1999. — ISBN 1-56592-479-7