ЧАСТЬ III

ЛАБОРАТОРНЫЕ РАБОТЫ

Редакция 31.03.2021

Лабораторная работа № 1

Изучение протокола SMTP (Simple Mail Transfer Protocol)

При выполнении заданий лабораторной работы рекомендуется ознакомиться с материалом части I учебного пособия "Передача данных в компьютерных сетях".

Задание

1. Изучить синтаксис команды **mail**, поставляемой в любом дистрибутиве Linux. Научиться формировать и отсылать почтовые сообщения с ее использованием.

2. С помощью утилиты **telnet** в терминальном режиме подключиться к 25-му TCP-порту любого компьютера, на котором запущен сервис smtp (например, к собственному, dims.karelia.ru, mail.petrsu.ru или mail.yandex.ru и др.) и, используя команды SMTP, отослать письмо на адрес, полученный у преподавателя (2017 год: **stud.labs@gmail.com**).

При этом соблюсти следующие требования:

а) в качестве обратного (своего) адреса указать адрес вида name@spam.ru, где вместо name указан псевдоним отсылающего;

б) в заголовках письма указать кодировку, в которой вы будете писать тело письма (например, KOI8-r, используемой по умолчанию в Linux), а также любой свой легальный адрес, на который якобы пойдет копия этого письма (у преподавателя должна сохраниться возможность узнать, куда была послана копия);

в) в теме (subject:) письма указать "SMTP, name" (без кавычек), где вместо name указана фамилия английскими буквами;

г) в теле письма:

- в первой строчке написать Фамилию Имя и Отчество русскими буквами и указать номер группы (при невозможности создать сообщение из русских символов можно попробовать воспользоваться вместо **telnet** программой **nc**, имеющей подобное назначение);
- во второй строчке указать IP-адрес компьютера, с которого осуществлялась отправка сообщения;
- в третьей строчке написать команду (блок команд), с помощью которой в bash (Linux) можно отправить по электронной почте в качестве тела письма содержимое какого-нибудь файла (для этого необходимо перенаправить содержимое этого файла на стандартный вход команды **mail**).

Примечания

1). Отсылка нескольких писем (например, в тестовом режиме от одного лица) на адрес, указанный преподавателем, является недопустимой.

2). Основное количество smtp-серверов обрабатывают адреса электронной почты при записи их в треугольных скобках <mail@mail.mail>.

Лабораторная работа № 2

Изучение протокола http (Hyper Text Transfer Protocol)

При выполнении заданий лабораторной работы рекомендуется ознакомиться с материалом части I учебного пособия "Передача данных в компьютерных сетях".

Задание

1. Изучить протокол **HTTP** (RFC 2616) (основные методы GET, POST, HEAD и схему клиент-серверного взаимодействия).

2. С помощью утилиты **telnet** в терминальном режиме подключиться к 80 TCP-порту компьютера, на котором запущен сервис httpd (например, thermo.karelia.ru или www.onego.ru) и, используя команды HTTP, проделать следующее:

а) сформировать такие запросы веб-серверу, чтобы получить в ответ коды 200, 400, 404 (описание кодов возврата дано в разделе 10 RFC 2616);

б) убедиться в том, что веб-сервер thermo.karelia.ru может отсылать shtml документы в архивированном виде, уменьшая исходящий трафик.

3. Получить любой документ с выбранного вами веб-сервера, запрашивая ресурс не напрямую, а через прокси-сервер proxy.karelia.ru (TCP-порт 81) (для компьютеров в компьютерном классе 125 УЛК-2 обратите внимание на IP-адрес прокси-сервера);

4. Написать отчет о проделанной работе, в котором привести доказательства проделанной работы по пунктам 2 и 3.

Лабораторная работа № 3

Исследование конфигурации сети университета и Карельского сегмента рунета

При выполнении заданий лабораторной работы рекомендуется ознакомиться с материалом части I учебного пособия "Передача данных в компьютерных сетях".

Задание

1. Научиться пользоваться командой **ping** (опции -t, -s, -c, -f, -i).

Исследовать разницу во времени доступа до компьютеров с доменными именами lab127.karelia.ru, iq.karelia.ru, thermo.karelia.ru, dfe3300.karelia.ru, dims.karelia.ru, plasma.karelia.ru, www.karelia.ru, petrsu.karelia.ru. Сделать предположения о физическом расположении данных серверов, а также пропускной способности каналов до них.

Узнать IP-адреса локальных интерфейсов можно командой **ifconfig**, получить информацию с DNS-серверов по командам **host**, **dig**, а также пользуясь сервисом **nslookup**.

2. Изучить интерфейс команды **traceroute** (tracert в Windows).

Протрассировать путь извне в Петрозаводск, пользуясь веб-трассировщиками с http://www.traceroute.org/. Сравнить прямой и обратные пути трассировщика между двумя хостами. При сравнении ориентироваться на IP-адреса и номера сетей маршрутизаторов.

Используя сервис **whois**, например на сайтах:

http://www.ripn.net/nic/whois/,
http://www.arin.net/whois/,
http://www.leader.ru/secure/,

узнать географическое положение промежуточных маршрутизаторов. Нанести отметки на схематически изображенную карту мира (карту России) (для отчета можно воспользоваться поиском картинок в www.google.com, набрав в строке поиска "worldmap" или что-то подобное). Построить географический путь прохождения пакетов до конечного пункта.

Примечание: маршрутизатор обычно имеет как минимум два IP-адреса по числу сетевых интерфейсов (сетевых карт); при исследовании сети вы будете видеть только один, ближний к вам.

3. Исследовать пропускную способность канала между двумя соседними маршрутизаторами, например Петрозаводском и Санкт-Петербургом (сеть Runet), с помощью утилит **ping** и **traceroute**. Для этого необходимо протрассировать путь до удаленного IP-адреса, выбрать два соседних маршрутизатора на пути следования пакета (желательно с разным временем отклика), несколько раз запустить утилиту **ping**, исследуя время отклика до каждого из выбранных маршрутизаторов, изменяя длину ICMP-эхо-пакета, узнать минимальное время обращения ICMP-пакета для каждого случая, построить график зависимости минимального времени обращения ICMP-пакета от его длины, вычислить пропускную способность по углу наклона и отсечке по оси времени. По разнице в скорости доступа (пропускной способности) до двух маршрутизаторов сделать вывод о пропускной способности канала между ними.

4. С помощью программы **traceroute** попытаться исследовать схему соединения маршрутизаторов в пределах Карелии, трассируя компьютеры из разных сетей (внутри университета или при исследовании сетей провайдеров sampo.ru, onego.ru, drevlanka.ru), построить в виде дерева схему соединения IP-сетей, включающую как минимум 3 маршрутизатора или 5 IP-сетей. Можно сравнить, например, пути до сетей 10.0.1.0/24 и 10.0.2.0/24 (последний байт в номере сети отведен под номер хоста, следовательно, при использовании в качестве параметра к traceroute этот байт должен быть ненулевым).

5. (факультативно) Написать программу (командный файл bash), которая в качестве входного параметра, введенного в командной строке, принимает номер сети (старшие три байта, разделенные и оканчивающиеся точками), перебирает значения младшего (в диапазоне 2–253), исследует количество строчек (промежуточных маршрутизаторов) в ответе **traceroute**, анализирует предпоследнюю строчку и выводит список всех последних по пути следования маршрутизаторов (соответственно, не больше 252 штук; повторы, то есть встречавшиеся ранее IP-адреса одного из интерфейсов маршрутизаторов, не должны быть выведены на экран). Таким образом, если последний по пути следования маршрутизатор всегда один и тот же, то все хосты, скорее всего, географически близко расположены и находятся в пределах одной IP-сети, то есть она не разбита на подсети.

В программе можно использовать команду **tail**, а также следует обратить внимание на то, что часть строчек **traceroute** выводит в stdout, а часть – в stderr. Для перенаправления стандартного вывода ошибок в канал stdout

использовать запись **2>&1**. Сформировав массив из слов, возвращаемых программой **traceroute**, сравнить IP-адрес искомого маршрутизатора с запомненным значением в предыдущей итерации цикла. В случае несовпадения вывести на экран новый IP-адрес.

Возможны и другие алгоритмы данной программы.

6. На основе полученных данных создать письменный отчет по всем выполненным пунктам.

Примечание: для того чтобы воспользоваться результатом выполнения упомянутых выше команд, в некоторых случаях необходимо будет воспользоваться средствами удаленного копирования файлов **sftp** и **scp**, реализованных в **ssh** (secure shell).

Лабораторная работа № 4

Сетевое программирование с использованием raw sockets

При выполнении заданий лабораторной работы рекомендуется ознакомиться с материалом части I учебного пособия "Передача данных в компьютерных сетях" (глава 9).

Задание

1. Постараться узнать изготовителя сетевой платы Ethernet своего компьютера по MAC-адресу интерфейса.

2. Научиться пользоваться программой перехвата сетевого трафика **tcpdump**. Для этого изучить синтаксис команды, после чего запустить сетевое приложение (браузер, утилиты **ping** или **traceroute**) и "раскодировать" (с помощью опции **–w**) содержимое Ethernet-кадров, отправленных к серверу и полученных от сервера доменных имен DNS (порт 53), доказать факт общения компьютера с сервером DNS (в презентациях к первой и последней лекциям продемонстрировано, как это сделать).

3. Написать программу, формирующую с использованием raw sockets TCP-сегмент, отправленный на определенный нелокальный IP-адрес, доказать факт посылки Ethernet-кадра путем анализа перехваченного трафика на компьютере-адресате.

При этом учесть, что:

- Номер TCP-порта получателя необходимо рассчитать на основании даты своего рождения. Дату рождения представить в формате целого числа YYYYMMDD (например, 19880230), далее преобразовать его в двухбайтовое целое беззнаковое число, например отсечением старших 16 битов. Алгоритм преобразования должен быть уникальным в пределах одной группы студентов и должен быть реализован в коде создаваемой программы.
- Флаги TCP-сегмента (URG, SYN, RST, ACK, FIN, PSH) необходимо выставить на основании младших шести битов числа, сформированного из месяца и числа даты рождения. Например, если дата рождения 25 марта (0325=145h), то младшие 6 битов – 000101b (145h && 3Fh = 5). Следовательно, нужно выставить флаги ACK и PSH. Данную процедуру необходимо организовать на языке программирования.
- Поскольку данная программа не предполагает установления TCP-соединения, необходимо "испортить" контрольную сумму в заголовках TCP-уровня, подставив в это поле значение, увязанное с датой рождения.

4. Создать отчет в письменной форме по проделанной работе. В отчете в качестве доказательств выполненной работы привести побайтовую расшифровку отдельных кадров захваченного трафика.

Примечание.
Выполнять задание можно удаленно. Для этого выделен сервер mars.phys.petrsu.ru, авторизация на который, - по обычному логину-паролю @dims.prv. Сервер mars.phys.petrsu.ru имеет IP-адрес 172.20.180.10. Доступ к нему возможен только внутри локальной сети ПетрГУ. Для того чтобы подключиться к нему извне (например, из сети rostelecom), необходимо сначала установить сеанс SSH-связи с сервером saturn.phys.petrsu.ru (у него белый IP-адрес), а потом с него, опять же в консольном режиме, подключиться к mars.phys.petrsu.ru.

Инструкции по работе с SSH под Linux и Windows приведены на странице "Инфо"/"SSH-шлюзы" (https://kompot.petrsu.ru/).

Утилиту перехвата сетевого трафика **tcpdump** можно запустить на локальных машинах в аудитории 110 УЛК-6, на сервере mars.phys.petrsu.ru и сервере с ip-адресом 172.20.175.60. При этом необходимо использовать команду **sudo** и указывать полный путь к исполняемому файлу утилиты:

```
sudo /usr/sbin/tcpdump
```

Если созданная программа исполняется на сервере mars.phys.petrsu.ru, то анализировать трафик необходимо либо на локальной машине в аудитории 110 УЛК-6, либо удаленно, т.е. на сервере с ip-адресом 172.20.175.60.

MAC-адреса, IP-адреса, а также другие настройки всех сетевых интерфейсов можно узнать, выполнив

```
/sbin/ifconfig
```

За основу создаваемой программы можно взять код, приведенный в пособии "Передача данных в компьютерных сетях".
Компилировать и запускать созданную программу отсылки TCP-сегмента обязательно из локальной файловой системы компьютера mars.phys.petrsu.ru (то есть из каталогов /tmp или в /home/localuser). Доступ к нему можно получить по протоколу ssh (возможно с помощью утилит scp и sftp).

Компилировать программу необходимо с ключом –Wall, чтобы компилятор предупреждал о всех возможных типах ошибок, пусть даже неявных и исправляемых компилятором.

gcc -Wall -o <exec_name> <source_name.c>

Необходимо откорректировать код программы таким образом, чтобы компилятор не выдавал ошибок (warning'ов), либо выдавал только системные, например, "_BSD_SOURCE and _SVID_SOURCE are deprecated, use _DEFAULT_SOURCE". Ошибок с преобразованием типов данных быть не должно.

Чтобы сделать файл исполняемым, необходимо правильно установить атрибуты файла в файловой системе: права доступа (например, утилитой chmod), а также одну из его способностей (capabilities) – CAP_NET_RAW. Последнее можно сделать с помощью созданного администратором скрипта /usr/local/bin/enable_netraw, запускаемого через sudo:

sudo enable_netraw имя_исполняемого_файла

Лабораторная работа № 5

Анализатор сетевого трафика на основе библиотеки pcap

При выполнении заданий лабораторной работы рекомендуется ознакомиться с материалом части I учебного пособия "Передача данных в компьютерных сетях" (глава 9).

Задание

Целью данной работы является изучение средств перехвата сетевых кадров на примере библиотеки **`pcap`** и разработка на основе последней простого анализатора данных, передаваемых по сетям на канальном, сетевом и более высоких уровнях модели OSI/RM.

1. Изучить интерфейс библиотеки для перехвата сетевых пакетов `pcap` (см. также `man pcap`).

2. Разработать и отладить скелет программы-перехватчика пакетов. Программа должна реализовать цикл перехвата пакетов и вывод информации о факте получения пакета. За основу создаваемой программы-перехватчика пакетов можно взять код, приведенный в пособии "Передача данных в компьютерных сетях" и осуществляющий перехват одного пакета (пункт 9.2, с. 135-136).

3. Получить у преподавателя задание по вариантам:

1) анализ распределения Ethernet-кадров по типу инкапсулированных данных;
2) анализ распределения Ethernet-кадров по длине кадра;
3) анализ распределения IP-датаграмм по размеру;
4) анализ распределения IP-датаграмм по значению поля TTL (Time-To-Live);
5) анализ распределения IP-датаграмм по типу инкапсулированных данных;
6) анализ распределения IP-датаграмм по адресам получателя;
7) анализ распределения IP-датаграмм по адресам отправителя;
8) анализ распределения IP-датаграмм по длине заголовочной части пакета;
9) анализ распределения IP-датаграмм по контрольной сумме (первый байт контрольной суммы);
10) анализ распределения исходящих IP-датаграмм по IP-адресам;
11) анализ распределения исходящих IP-датаграмм по парам MAC-адрес – IP-адрес;
12) анализ распределения исходящих IP-датаграмм по контрольной сумме (последний байт контрольной суммы);
13) анализ распределения ICMP-сообщений по типам;
14) анализ распределения ICMP-сообщений по размеру ICMP-пакета;
15) анализ распределения TCP-сегментов по порту назначения;
16) анализ распределения TCP-сегментов по порту источника;
17) анализ распределения TCP-сегментов по выставленным флагам (URG, ACK, PSH, RST, SYN, FIN);
18) анализ распределения TCP-сегментов по размеру окна для разных приложений;
19) анализ распределения TCP-сегментов по размеру;
20) анализ распределения исходящих TCP-сегментов по порту источника;
21) анализ распределения UDP-пакетов по порту назначения;
22) анализ распределения UDP-пакетов по порту источника;
23) анализ распределения UDP-пакетов по значению контрольной суммы (старший байт контрольной суммы);
24) анализ распределения DNS-пакетов по типу (запрос / ответ);
25) анализ распределения DNS-ответов по длине;
26) анализ временного распределения входящего Ethernet-трафика;
27) анализ временного распределения исходящего Ethernet-трафика;
28) анализ временного распределения широковещательного Ethernet-трафика;
29) анализ временного распределения ARP-запросов;

30) анализ временного распределения ARP-ответов;
31) анализ временного распределения не IP-трафика;
32) анализ временного распределения исходящих широковещательных IP-датаграмм;
33) анализ временного распределения входящих IP-датаграмм;
34) анализ временного распределения DNS-запросов;
35) анализ временного распределения DNS-ответов;
36) анализ временного распределения ICMP-пакетов;
37) анализ временного распределения TCP-сегментов с флагами PSH или URG;
38) анализ временного распределения TCP-сегментов с флагом FIN;
39) анализ процентного содержания TCP-сегментов во всех IP-датаграммах;
40) анализ процентного содержания UDP-датаграмм во всех IP- датаграммах.

4. В соответствии с выбранным вариантом модифицировать разработанный по пункту 2 перехватчик таким образом, чтобы он производил тот или иной анализ (!) сетевых пакетов. Это значит, что разработанная программа должна не только перехватывать трафик, но и предварительно его обработать. Например, сформировать массив из 100 элементов, в значения которых записаны количества пакетов, полученных последовательно каждую секунду в течение 100 секунд. Построение гистограммы на основе значений элементов этого массива допускается в стороннем ПО, например, в MS Office. Фильтр pcap, осуществляющий выборку пакетов, должен соответствовать выбранному варианту задания. Корректность работы фильтра можно предварительно проверить с помощью программы перехвата сетевого трафика tcpdump.

5. Продемонстрировать работу анализатора преподавателю.

6. Сохранить результат работы анализатора в файл и по содержащимся в нем данным построить диаграмму (в зависимости от варианта задания), показать диаграмму преподавателю, создать письменный отчет по проделанной работе.

Примечания.

1. Во время работы анализатора обязательно (!) загрузить сетевой работой узел, подвергающийся прослушиванию (запускать различные сетевые приложения, пинговать его с других компьютеров в сети).

2. MAC-адреса, IP-адреса, а также другие настройки всех сетевых интерфейсов можно узнать, выполнив
/sbin/ifconfig

3. Компилировать программу необходимо с ключом –Wall, чтобы компилятор предупреждал обо всех возможных типах ошибок, пусть даже неявных и исправляемых компилятором, а также с ключом –lpcap, для корректного подключения функций библиотеки pcap в исполняемое приложение:

gcc -Wall -o <exec_name> <source_name.c> -lpcap

Запускать созданную программу можно из локальной файловой системы компьютера mars.phys.petrsu.ru (то есть из каталогов /tmp или в /home/localuser).
Доступ к нему можно получить по протоколу ssh (возможно с помощью утилит scp и sftp).
Чтобы сделать файл исполняемым, необходимо правильно установить атрибуты файла в файловой системе: права доступа (например, утилитой chmod), а также одну из его способностей (capabilities) – CAP_NET_RAW. Последнее можно сделать с помощью созданного администратором скрипта /usr/local/bin/enable_netraw, запускаемого через sudo:

sudo enable_netraw имя_исполняемого_файла

Лабораторная работа № 6

Создание плана-схемы локального сегмента сети

Задание

1. Найти локальный сегмент сети (например, в домашней сети), удовлетворяющий следующим требованиям:

а) построен на основе беспроводной точка доступа и/или коммутатора/маршрутизатора;
б) содержит не менее двух компьютеров и одного беспроводного мобильного устройства (смартфон, ноутбук, Bluetooth-устройство, др.).

2. Создать план-схему этого сегмента сети.

Для создания рисунка сети можно использовать одну из свободно распространяемых программ, например, gliffy (https://www.gliffy.com/products/online/) или из списка на странице http://www.techrepublic.com/blog/five-apps/five-free-apps-for-diagramming-your-network/ .

На схеме необходимо обозначить способы сетевого подключения устройств: тип интерфейса, тип кабеля, номер порта, маркировку розеток (если имеются), а также для каждого сетевого интерфейса каждого устройства в сети необходимо обозначить модель, MAC-адрес или другой адрес канального уровня по модели OSI/RM (по возможности), IP-адрес (по возможности), адрес прикладного уровня по модели OSI/RM (по возможности).

На схеме обязательно должен быть обозначен как минимум 1 маршрутизатор. Желательно обозначить сетевые принтеры, файрвол, NAT-сервер, телевизор и другие объекты сетевой инфраструктуры с указанием их ключевых сетевых настроек.

Для поиска необходимой информации можно пользоваться утилитами **arp**, **netstat**, **route**, **ping**, **traceroute**, веб-сервисами удаленной трассировки, а также изучить журнал домашней точки доступа / маршрутизатора.

3. На одном из компьютеров в локальной сети проделать следующее.
а) Добиться изменений в arp-таблице. Для этого получить arp-таблицу (команда **arp**), изучить ее. Затем посовершать какие-нибудь действия в сети (например, пропинговать соседние узлы в локальной сети), затем снова получить новую таблицу с изменениями.
б) Получить таблицу маршрутизации (команда **route**). Изучить ее структуру.
в) Добиться изменений в таблице текущих соединений. Для этого необходимо получить список открытых соединений (команда **netstat**), изучить ее. Затем посовершать какие-нибудь действия в сети (например, установить новое соединение с веб-сервером с помощью браузера или установить сеанс связи по протоколу SMTP), затем снова получить новую таблицу с изменениями.

4. Написать отчет о проделанной работе. В него включить информацию, полученную в ходе работ по пунктам 2 и 3а-3в. Кроме приложенной плана-схемы сегмента сети в отчете необходимо указать, каким образом была найдена информация о сетевых настройках устройств. По всем работам в пунктах 3а-3в необходимо проанализировать результаты и сделать выводы.

Примечание: штрафные баллы начисляются за неполноту представленной информации.