

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра теоретической прикладной информатики

Лабораторная работа № 5
по дисциплине «Операционные системы, среды и оболочки»

Анализ функционирования и диагностика IP-сетей

Факультет:	ПМИ
Группа:	ПМ-92
Бригада:	8
Студенты:	Иванов В., Кутузов И.
Преподаватель:	Кобылянский В. Г.

Новосибирск

2021

Цель работы

Приобретение практических навыков работы с сетевыми командами операционных систем Windows и Linux, предназначенными для анализа и диагностики сетей TCP/IP, а также со средствами интерактивной диагностики сетей.

Задание

Первый этап

Изучить основные способы диагностики IP-сетей. Задания выполняются на рабочем компьютере (ПК) с установленной ОС Windows и на удаленных серверах.

Второй этап

Реализовать **Linux**-приложение, которое будет выполнять функцию запроса маски сети утилиты **ping**.

Примечание: запрос ICMP Address Mask является устаревшим и не поддерживается большинством маршрутизаторов. В разработанной программе реализован **запрос эхо-повтора**.

Ход работы

1. Подключиться с помощью клиента Putty к серверу fpm2.ami.nstu.ru и с помощью команды **uname** получить полную информацию об установленной операционной системе и аппаратной платформе.

Работу будем выполнять на компьютере с ОС Linux, подключившись к удаленному рабочему столу ОС Windows, работающему на базе Облачной платформы НГТУ, при помощи клиента **Remmina**. В дальнейшем этот рабочий стол будем называть рабочим компьютером (ПК), а соединение с сервером fpm2.ami.nstu.ru установим через ОС Linux.

Подключимся к серверу fpm2.ami.nstu.ru при помощи **ssh**:

```
~ ➤ ssh pmi-b9208@fpm2.ami.nstu.ru
pmi-b9208@fpm2.ami.nstu.ru's password:
Last login: Tue Nov 16 19:07:04 2021 from 212.164.64.2
[pmi-b9208@students ~]$ |
```

Получим информацию об установленной ОС и аппаратной платформе:

```
[pmi-b9208@students ~]$ uname -a
Linux students.ami.nstu.ru 3.10.0-327.3.1.el7.x86_64
4 #1 SMP Wed Dec 9 14:09:15 UTC 2015 x86_64 x86_64
x86_64 GNU/Linux
```

2. Получить статистику по сетевым интерфейсам ПК и сервера fpm2.ami.nstu.ru, пояснить результаты.

fpm2.ami.nstu.ru

```
[pmi-b9208@students ~]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 217.71.130.131 netmask 255.255.255.128 broadcast 217.71.130.255
    inet6 fe80::215:5dff:fe82:8d01 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:82:8d:01 txqueuelen 1000 (Ethernet)
    RX packets 188776481 bytes 26471644058 (24.6 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 39184175 bytes 45326944199 (42.2 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 0 (Local Loopback)
    RX packets 31048575 bytes 19122344220 (17.8 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 31048575 bytes 19122344220 (17.8 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:d4:60:b6 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
[pmi-b9208@students ~]$ netstat -i
Kernel Interface table
Iface      MTU      RX-OK RX-ERR RX-DRP RX-OVR      TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500    189213768      0      0 0      39400368      0      0      0 BMRU
lo         65536   31418747      0      0 0      31418747      0      0      0 LRU
virbr0     1500      0      0      0 0          0      0      0      0 BMU
```

Всего имеется 3 сетевых интерфейса: **eth0** (физический адаптер, используется для выхода в сеть Интернет), **lo** (loopback-интерфейс, используется только на данном устройстве) и **virbr0** (интерфейс виртуального моста, предназначен для виртуальных машин).

В первой строке информации о каждом сетевом интерфейсе (а также в столбце **Flg** таблицы, полученной командой **netstat -i**) указаны флаги.

Флаг статуса интерфейса **UP** (DOWN) говорит о том, что интерфейс в данный момент (не) инициализирован.

Флаг **BROADCAST** говорит о поддержке интерфейсом широковещательной рассылки IPv4.

Флаг статуса передачи **RUNNING** говорит о том, что интерфейс в данный момент передает пакеты.

Флаг **MULTICAST** указывает на поддержку мультивещания (групповой рассылки).

Рассмотрим теперь характеристики сетевых интерфейсов:

mtu - максимальный размер передаваемых пакетов в октетах (байтах)

inet - IPv4-адрес, назначенный интерфейсу

inet6 - IPv6-адрес, назначенный интерфейсу

netmask - маска IPv4 подсети

broadcast - широковещательный IPv4-адрес

prefixlen - длина префикса IPv6 подсети

scopeid - IPv6 scope, указывающий в какой части сети адрес действителен

ether - MAC-адрес интерфейса

txqueuelen - размер буфера передачи (отправка данных осуществляется при заполнении буфера до указанного значения)

RX packets и **TX packets** - количество полученных и переданных пакетов/байтов соответственно

RX errors и **TX errors** - статистика по полученным и переданным пакетам соответственно; **dropped (DRP)** - количество пропавших пакетов, **overruns (OVR)** - количество ошибок из-за превышения скорости.

Заметим, что **IPv6**-адреса используют не маски, а префиксы. Они выполняют ту же функцию, что и маски - отделяют адрес подсети от адреса хоста (интерфейса). Например, длина префикса /64 указывает на то, что первые 64 бита **IPv6**-адреса относятся к адресу сети, а оставшиеся 64 - к адресу хоста (интерфейса).

Также в **IPv6** нет широковещательного адреса (он был заменен на ANYCAST).

Обратим также внимание на характеристики **loopback**-интерфейса. Любые сообщения, посылаемые на этот канал принимаются им же самим. В связи с этим, значения **RX** и **TX** совпадают, а **буфер передачи** имеет нулевой размер.

ПК

Примечание: терминальный сервер НГТУ использует технологию **FlexibleLOM** - встроенный в материнскую плату четырехпортовый чипсет. Поскольку интерфейсы этого типа отличаются лишь MAC-адресом, в отчет включен только первый из них (**Адаптер Ethernet Embedded FlexibleLOM 1 Port 2**).

```
C:\Users\v.v.ivanov.2019>ipconfig /all
```

Настройка протокола IP для Windows

```
Имя компьютера . . . . . : cloud-rdhost1
Основной DNS-суффикс . . . . . : corp.nstu.ru
Тип узла. . . . . : Гибридный
IP-маршрутизация включена . . . . : Нет
WINS-прокси включен . . . . . : Нет
Порядок просмотра суффиксов DNS . : corp.nstu.ru
                                         local
```

Адаптер Ethernet Embedded FlexibleLOM 1 Port 2:

```
Состояние среды. . . . . : Среда передачи недоступна.
DNS-суффикс подключения . . . . . :
Описание. . . . . : HPE Ethernet 1Gb 4-port 366FLR Adapter
Физический адрес. . . . . : 48-DF-37-C8-9E-A1
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
```

Адаптер Ethernet nTeam:

```
DNS-суффикс подключения . . . . . :  
Описание. . . . . : Microsoft Network Adapter Multiplexor Driver  
Физический адрес. . . . . : 3C-FD-FE-AD-65-08  
DHCP включен. . . . . : Нет  
Автонастройка включена. . . . . : Да  
Локальный IPv6-адрес канала . . . : fe80::df7:61bf:1c71:67b1%8(Основной)  
IPv4-адрес. . . . . : 172.17.1.18(Основной)  
Маска подсети . . . . . : 255.255.0.0  
Основной шлюз. . . . . : 172.17.1.1  
IAID DHCPv6 . . . . . : 356318718  
DUID клиента DHCPv6 . . . . . : 00-01-00-01-27-D4-10-A8-48-DF-37-C8-9E-A1  
DNS-серверы. . . . . : 217.71.129.140  
                        217.71.129.162  
NetBios через TCP/IP. . . . . : Включен
```

Адаптер Ethernet Ethernet:

```
DNS-суффикс подключения . . . . . : local  
Описание. . . . . : Generic USB-EEM Network Adapter  
Физический адрес. . . . . : 92-BC-7C-DB-2E-AB  
DHCP включен. . . . . : Да  
Автонастройка включена. . . . . : Да  
Локальный IPv6-адрес канала . . . : fe80::6076:c9f0:ac5d:c4da%11(Основной)  
IPv4-адрес. . . . . : 16.1.15.2(Основной)  
Маска подсети . . . . . : 255.255.255.252  
Аренда получена. . . . . : 5 октября 2021 г. 3:14:07  
Срок аренды истекает. . . . . : 13 января 2022 г. 3:14:52  
Основной шлюз. . . . . :  
DHCP-сервер. . . . . : 16.1.15.1  
IAID DHCPv6 . . . . . : 127057020  
DUID клиента DHCPv6 . . . . . : 00-01-00-01-27-D4-10-A8-48-DF-37-C8-9E-A1  
DNS-серверы. . . . . : fec0:0:0:ffff::1%1  
                        fec0:0:0:ffff::2%1  
                        fec0:0:0:ffff::3%1  
NetBios через TCP/IP. . . . . : Включен
```

Работающие с сетью Интернет интерфейсы можно определить по наличию у них IPv4-адреса и маски подсети. Все интерфейсы имеют MAC-адрес, информацию о состоянии и автонастройке протокола **DHCP**, а также информацию о **DHCPv6**: **IAID** (интерфейс клиентской системы) и **DUID** (идентификатор клиентской системы, используется клиентом для получения IP-адреса от DHCPv6-сервера). Также имеется информация о системе доменных имен **DNS**: DNS-серверы, DNS-суффиксы подключения и порядок их просмотра.

Получим статистику Ethernet:

```
C:\Users\v.v.ivanov.2019>netstat -e  
Статистика интерфейса
```

	Получено	Отправлено
Байт	3404450804	3875315344
Одноадресные пакеты	3092252812	3125978536
Многоадресные пакеты	24665868	246344
Отброшено	0	0
Ошибки	0	6
Неизвестный протокол	0	

Таблица показывает количество переданных и полученных байтов и пакетов, а также их типы и информацию об ошибках при передаче/получении.

3. Просмотреть содержимое DNS-кэша, пояснить характеристики записей, очистить кэш.

Система доменных имен используется для получения информации о доменах. Кэш DNS содержит информацию обо всех последних посещениях (и попытках посещения) веб-сайтов, а также другие IP-адреса сайтов. Это позволяет быстрее загружать ранее посещенные страницы.

На скриншоте видим, что кэш состоит из записей, имеющих имена, - это домены, с которыми устанавливалось соединение. **Тип записи** - это формат и назначение записи: наиболее популярными являются **A (1)** - соответствие IPv4 адреса доменному имени, **AAAA (28)** - IPv6 и **CNAME (5)** - привязка псевдонима (например, `www.example.com` и `example.com`). На скриншоте видны только записи типа **A**, поэтому для них выводится **A-запись** (IPv4-адрес домена), а **длина данных** соответствует длине IPv4-адреса.

Очистка DNS-кэша производится командой `ipconfig /flushdns`, однако на удаленном рабочем столе это невозможно ввиду отсутствия прав администратора.

```
C:\Users\v.v.ivanov.2019>ipconfig /displaydns
```

Настройка протокола IP для Windows

safebrowsing.googleapis.com

Нет записей типа AAAA

safebrowsing.googleapis.com

Имя записи. : safebrowsing.googleapis.com

Тип записи. : 1

Срок жизни. : 266

Длина данных. : 4

Раздел. : Ответ

A-запись (узла) . . . : 173.194.222.95

Имя записи. : f.root-servers.net

Тип записи. : 1

Срок жизни. : 266

Длина данных. : 4

Раздел. : Дополнительно

A-запись (узла) . . . : 192.5.5.241

Первые несколько записей DNS-кэша

4. Просмотреть содержимое ARP-таблицы, пояснить характеристики записей, выполнить добавление и удаление статических записей.

ПК

ARP-таблица показывает соответствия IP-адресов с MAC-адресами, а также их тип.

```
C:\Users\v.v.ivanov.2019>arp -a
```

Интерфейс: 172.17.1.18 --- 0x8

адрес в Интернете	Физический адрес	Тип
172.17.1.1	00-1a-4a-16-03-e7	динамический
172.17.1.10	00-1a-4a-16-01-28	динамический
172.17.1.11	00-1a-4a-16-01-07	динамический

Первые записи ARP-таблицы

fpm2.ami.nstu.ru

```
[pmi-b9208@students ~]$ arp
```

Address	HWtype	HWaddress	Flags Mask	Iface
gw-130-208v.ami.nstu.ru	ether	00:21:1b:f5:76:46	C	eth0
screamer.ami.nstu.ru	ether	f8:32:e4:89:1a:8d	C	eth0
gw-130-204.ami.nstu.ru	ether	00:21:1b:ee:dd:c4	C	eth0
fpm.ami.nstu.ru	ether	00:1e:0b:d9:84:48	C	eth0
ksc.ami.nstu.ru	ether	00:15:5d:82:8f:1b	C	eth0
pmt-08.ami.nstu.ru	ether	00:15:5d:82:8d:bf	C	eth0
armor.ami.nstu.ru	ether	2c:4d:54:51:91:59	C	eth0
gw-130.ami.nstu.ru	ether	00:00:0c:9f:f0:82	C	eth0
217.71.130.146	ether	00:15:5d:82:b0:01	C	eth0
gate.ami.nstu.ru	ether	d4:8c:b5:4d:f6:5b	C	eth0

В столбце **Flags Mask** указан тип записей - динамический, в столбце **Iface** указан сетевой интерфейс, который использует эти узлы.

Добавление и удаление записей невозможны ввиду отсутствия прав администратора (как на удаленном рабочем столе, так и на сервере fpm2.ami.nstu.ru). Добавленные в таблицу адреса будут иметь статический тип.

5. Просмотреть содержимое таблицы маршрутизации, пояснить характеристики записей.

ПК

В таблице представлены **сетевые адреса** различных узлов сети, с помощью которых маршрутизатор определяет, куда отправлять IP-пакет.

Столбец **Адрес шлюза** задает IP-адрес следующего ближайшего маршрутизатора (таким образом пакет продвигается к сети назначения).

Значение **On-link** говорит о том, что адрес доступен напрямую (не требуются доп. маршрутизаторы).

Метрика используется для выбора маршрута (чем меньше, тем маршрут предпочтительнее).

Столбец **Интерфейс** содержит IP-адрес выходного сетевого интерфейса данного маршрутизатора.

```
C:\Users\v.v.ivanov.2019>route print
```

Список интерфейсов

```
17...48 df 37 c8 9e a1 .....HPE Ethernet 1Gb 4-port 366FLR Adapter
5...48 df 37 c8 9e a2 .....HPE Ethernet 1Gb 4-port 366FLR Adapter #2
12...48 df 37 c8 9e a0 .....HPE Ethernet 1Gb 4-port 366FLR Adapter #3
19...48 df 37 c8 9e a3 .....HPE Ethernet 1Gb 4-port 366FLR Adapter #4
8...3c fd fe ad 65 08 .....Microsoft Network Adapter Multiplexor Driver
11...92 bc 7c db 2e ab .....Generic USB-EEM Network Adapter
1.....Software Loopback Interface 1
```

IPv4 таблица маршрута

Активные маршруты:

Сетевой адрес	Маска сети	Адрес шлюза	Интерфейс	Метрика
0.0.0.0	0.0.0.0	172.17.1.1	172.17.1.18	271
16.1.15.0	255.255.255.252	On-link	16.1.15.2	281
255.255.255.255	255.255.255.255	On-link	172.17.1.18	271

Постоянные маршруты:

Сетевой адрес	Маска	Адрес шлюза	Метрика
0.0.0.0	0.0.0.0	172.17.1.1	По умолчанию

IPv6 таблица маршрута

Активные маршруты:

Метрика	Сетевой адрес	Шлюз
1	331 ::1/128	On-link
11	281 fe80::/64	On-link
8	271 fe80::/64	On-link
8	271 fe80::df7:61bf:1c71:67b1/128	On-link
11	281 fe80::6076:c9f0:ac5d:c4da/128	On-link
1	331 ff00::/8	On-link
11	281 ff00::/8	On-link
8	271 ff00::/8	On-link

Постоянные маршруты:

Отсутствует

fpm2.ami.nstu.ru

```
[pmi-b9208@students ~]$ route
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	gw-130.ami.nstu	0.0.0.0	UG	100	0	0	eth0
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	virbr0
217.71.130.128	0.0.0.0	255.255.255.128	U	100	0	0	eth0

Здесь столбец **Destination** является аналогом столбца Сетевой адрес, **Gateway** - аналогом столбца Интерфейс, **Metric** и **Genmask** - метрика и маска соответственно.

6. В командном режиме на ПК и на сервере определить IP-адреса поисковой системы **yahoo.ru**, пояснить результаты.

ПК

```
C:\Users\v.v.ivanov.2019>nslookup yahoo.ru
Server: ns1.cloud.nstu.ru
Address: 217.71.129.140
```

Не заслуживающий доверия ответ:

```

Name: yahoo.ru
Addresses: 212.82.100.150
           98.136.103.23
           74.6.136.150
```

fpm2.ami.nstu.ru

```
[pmi-b9208@students ~]$ nslookup yahoo.ru
Server: 217.71.130.130
Address: 217.71.130.130#53
```

Non-authoritative answer:

```

Name: yahoo.ru
Address: 212.82.100.150
Name: yahoo.ru
Address: 74.6.136.150
Name: yahoo.ru
Address: 98.136.103.23
```

На **ПК** видим IP-адрес **217.71.129.140** - это адрес DNS-сервера **ns1.cloud.nstu.ru**. Поскольку домен **yahoo.ru** является поисковой системой, для распределения нагрузки он использует несколько IP-адресов, указанных в поле **Addresses**.

На **fpm2.ami.nstu.ru** используемый DNS-сервер - **217.71.130.130**. IP-адреса поисковой системы совпадают.

7. В командном режиме на ПК и на сервере определить IP-адрес узлов сети **ethz.ch** и **wikimapia.org**, выполнить пингование и трассировку.

ПК

fpm2.ami.nstu.ru

ethz.ch

Определение IP-адреса


```
C:\Users\v.v.ivanov.2019>nslookup ethz.ch [pmi-b9208@students ~]$ nslookup ethz.ch
ᐅxËtxË: ns1.cloud.nstu.ru Server: 217.71.130.130
Address: 217.71.129.140 Address: 217.71.130.130#53

Не заслуживающий доверия ответ: Non-authoritative answer:
ᐅ : ethz.ch Name: ethz.ch
Address: 129.132.19.216 Address: 129.132.19.216
```

Пингование

```
C:\Users\v.v.ivanov.2019>ping ethz.ch [pmi-b9208@students ~]$ ping -c 4 ethz.ch
Обмен пакетами с ethz.ch [129.132.19.216] с 32 байтами данных: PING ethz.ch (129.132.19.216) 56(84) bytes of data.
Ответ от 129.132.19.216: число байт=32 время=117мс TTL=42 64 bytes from cms-publish.ethz.ch (129.132.19.216): icmp_seq=1 ttl=42 time=117 ms
Ответ от 129.132.19.216: число байт=32 время=117мс TTL=42 64 bytes from cms-publish.ethz.ch (129.132.19.216): icmp_seq=2 ttl=42 time=117 ms
Ответ от 129.132.19.216: число байт=32 время=117мс TTL=42 64 bytes from cms-publish.ethz.ch (129.132.19.216): icmp_seq=3 ttl=42 time=117 ms
Ответ от 129.132.19.216: число байт=32 время=117мс TTL=42 64 bytes from cms-publish.ethz.ch (129.132.19.216): icmp_seq=4 ttl=42 time=117 ms
```

Трассировка

```
C:\Users\v.v.ivanov.2019>tracert ethz.ch [pmi-b9208@students ~]$ traceroute ethz.ch
Трассировка маршрута к ethz.ch [129.132.19.216] traceroute to ethz.ch (129.132.19.216), 30 hops max, 60 byte packets
с максимальным числом прыжков 30: 1 gw-130-208v.ami.nstu.ru (217.71.130.251) 2.564 ms 2.810 ms 2.826 ms
2 * * *
3 * * *
28 * * *
29 * * *
30 * * *

1 <1 мс <1 мс <1 мс 172.17.1.1
2 * * * Превышен интервал ожидания для запроса.
3 <1 мс <1 мс <1 мс ciu-ferma-1lb-ii-gw.nstu.ru [217.71.128.56]
4 <1 мс <1 мс <1 мс mx10.nstu.ru [217.71.128.49]
5 <1 мс <1 мс <1 мс ttk-1-gw.nsk.runnet.ru [194.85.38.45]
6 50 ms 49 ms 50 ms vlan1154.msk-m9-3-gw.runnet.ru [194.85.40.65]
7 57 ms 57 ms 58 ms spb-bm18-2-gw.runnet.ru [194.85.40.205]
8 59 ms 59 ms 58 ms spb-kt12-2-gw.spb.runnet.ru [194.85.40.90]
9 64 ms 67 ms 64 ms fi-csc2.nordu.net [109.105.102.57]
10 78 ms 78 ms 78 ms de-hmb.nordu.net [109.105.97.77]
11 86 ms 87 ms 86 ms de-ffm.nordu.net [109.105.97.105]
12 97 ms 97 ms 97 ms uk-hex.nordu.net [109.105.97.78]
13 98 ms 98 ms 99 ms nordunet.mx1.lon.uk.geant.net [62.40.124.129]
14 99 ms 99 ms 99 ms ae6.mx1.lon2.uk.geant.net [62.40.98.37]
15 106 ms 106 ms 106 ms ae5.mx1.par.fr.geant.net [62.40.98.179]
16 112 ms 112 ms 112 ms ae5.mx1.gen.ch.geant.net [62.40.98.182]
17 112 ms 111 ms 112 ms swice1-100ge-0-3-0-1.switch.ch [62.40.124.22]
18 114 ms 113 ms 113 ms swice4-b4.switch.ch [130.59.36.70]
19 114 ms 114 ms 113 ms swibf1-b2.switch.ch [130.59.36.113]
20 117 ms 117 ms 117 ms swiez3-b5.switch.ch [130.59.37.6]
21 117 ms 118 ms 117 ms rou-gw-lee-tengig-to-switch.ethz.ch [192.33.92.1]
22 117 ms 117 ms 117 ms rou-fw-rz-rz-gw.ethz.ch [192.33.92.169]
23 117 ms 117 ms 117 ms cms-publish.ethz.ch [129.132.19.216]

Трассировка завершена.
```

wikimapia.org

Определение IP-адреса

```
C:\Users\v.v.ivanov.2019>nslookup wikimapia.org [pmi-b9208@students ~]$ nslookup wikimapia.org
ᐅxËtxË: ns1.cloud.nstu.ru Server: 217.71.130.130
Address: 217.71.129.140 Address: 217.71.130.130#53

Не заслуживающий доверия ответ: Non-authoritative answer:
ᐅ : wikimapia.org Name: wikimapia.org
Address: 88.99.95.180 Address: 88.99.95.180
```

Пингование

```
C:\Users\v.v.ivanov.2019>ping wikimapia.org [pmi-b9208@students ~]$ ping -c 4 wikimapia.org
Обмен пакетами с wikimapia.org [88.99.95.180] с 32 байтами данных: PING wikimapia.org (88.99.95.180) 56(84) bytes of data.
Ответ от 88.99.95.180: число байт=32 время=93мс TTL=50 64 bytes from www.wikimapia.org (88.99.95.180): icmp_seq=1 ttl=50 time=93.3 ms
Ответ от 88.99.95.180: число байт=32 время=93мс TTL=50 64 bytes from www.wikimapia.org (88.99.95.180): icmp_seq=2 ttl=50 time=93.4 ms
Ответ от 88.99.95.180: число байт=32 время=93мс TTL=50 64 bytes from www.wikimapia.org (88.99.95.180): icmp_seq=3 ttl=50 time=93.3 ms
Ответ от 88.99.95.180: число байт=32 время=93мс TTL=50 64 bytes from www.wikimapia.org (88.99.95.180): icmp_seq=4 ttl=50 time=93.3 ms
```

Трассировка

C:\Users\v.v.ivanov.2019>tracert wikiapia.org

Трассировка маршрута к wikiapia.org [88.99.95.180]
с максимальным числом прыжков 30:

```

 1  <1 ms    <1 ms    <1 ms    172.17.1.1
 2  *        *        *        Превышен интервал ожидания для запроса.
 3  <1 ms    <1 ms    <1 ms    ciu-ferma-lib-11-gw.nstu.ru [217.71.128.56]
 4  <1 ms    <1 ms    <1 ms    mx10.nstu.ru [217.71.128.49]
 5  1 ms     1 ms     1 ms     217.8.237.17
 6  1 ms     1 ms     1 ms     stn-cr03-be20.10.nsk.mts-internet.net [195.34.36.57]
 7  50 ms    50 ms    50 ms    stn-cr01-be3.54.nsk.mts-internet.net [195.34.50.188]
 8  50 ms    50 ms    50 ms    zoo-cr03-be8.66.ekt.mts-internet.net [212.188.42.149]
 9  50 ms    50 ms    50 ms    vish-cr01-be7.66.kaz.mts-internet.net [212.188.29.85]
10  50 ms    50 ms    50 ms    mag9-cr02-be6.16.msk.mts-internet.net [195.34.50.161]
11  88 ms    86 ms    93 ms    a197-cr01-ae10.77.msk.mts-internet.net [195.34.50.73]
12  83 ms    82 ms    83 ms    anc-cr03-ae3.77.ff.mts-internet.net [195.34.59.50]
13  88 ms    91 ms    88 ms    static.213.133.118.145.clients.your-server.de [213.133.118.145]
14  88 ms    91 ms    88 ms    core1.fra.hetzner.com [213.239.245.125]
15  92 ms    92 ms    93 ms    core24.fsn1.hetzner.com [213.239.229.78]
16  94 ms    94 ms    94 ms    ex9k1.dc1.fsn1.hetzner.com [213.239.245.234]
17  93 ms    94 ms    93 ms    www.wikiapia.org [88.99.95.180]

```

```


[pmi-b9208@students ~]$ traceroute wikiapia.org
traceroute to wikiapia.org (88.99.95.180), 30 hops max, 60 byte packets
 1  gw-130-208v.ami.nstu.ru (217.71.130.251)  7.691 ms  7.685 ms  7.858 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *

```


8. С помощью интерактивных сетевых сервисов выполнить трассировку, определить местонахождение и владельца узлов сети **ethz.ch** и **wikiapia.org**. Результат трассировки в виде скриншота географической карты представить в отчете и выполнить его анализ. Начальный пункт трассировки - г. Новосибирск.

Для начала определим местоположение и владельцев узлов.

ethz.ch находится в Цюрихе:

Domain Name	Country	Region	City
ethz.ch	Switzerland 	Zurich	Zurich
ISP	Organization	Latitude	Longitude
Swiss Federal Institute of Technology Zurich	Not Available	47.3667	8.5500

wikiapia.org находится в Фалькенштайне (Саксония):

Domain Name	Country	Region	City
wikiapia.org	Germany 	Saxony	Falkenstein
ISP	Organization	Latitude	Longitude
Hetzner Online GmbH	Hetzner	50.475	12.365

ISP - интернет-провайдер.

Теперь определим владельцев при помощи сервисов **lookup.icann.org** и **nic.ch**:

Name: GoDaddy.com, LLC

IANA ID: 146

Abuse contact email: abuse@godaddy.com

Abuse contact phone: tel:480-624-2505

Registrar

[Swizzonic AG](#) 

Postfach

CH-8021 Zürich

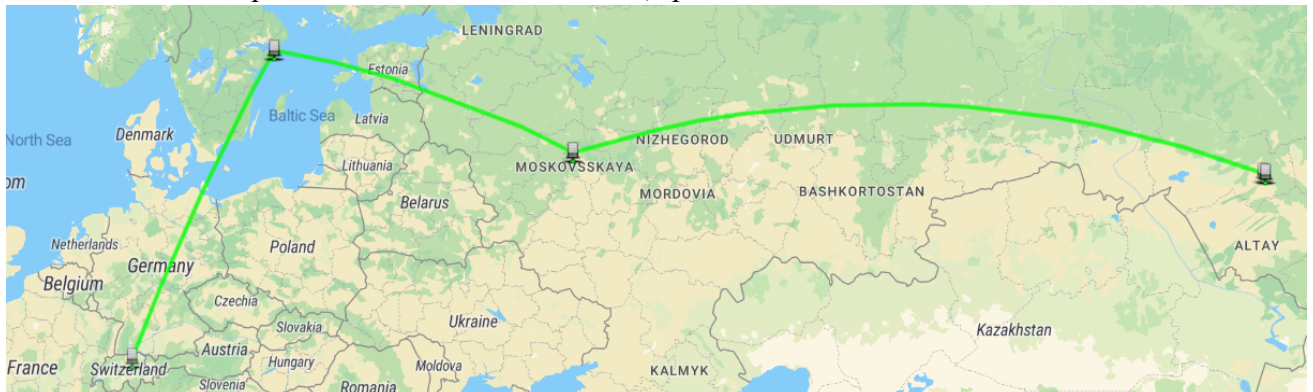
Phone +41 848696969

helpdesk@swizzonic.ch

Владелец ethz.ch - **Swizzonic AG**.
Владелец wikimapia.org - **GoDaddy.com, LLC**.

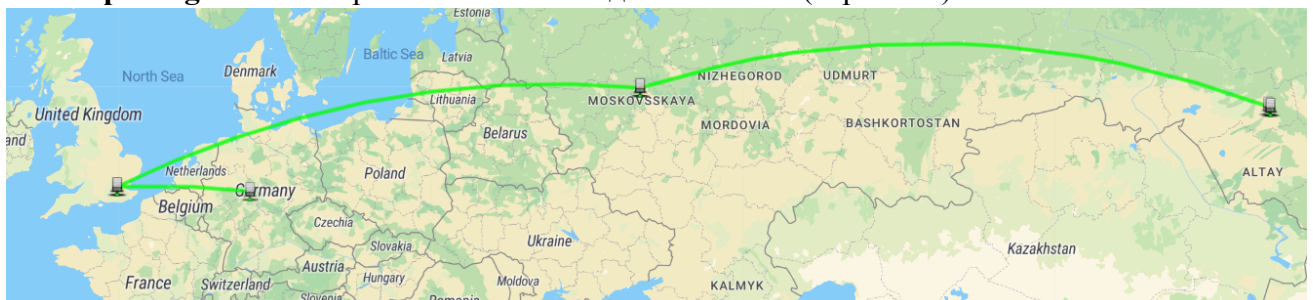
Выполним трассировку узлов при помощи сервиса **ping-admin.ru**:

ethz.ch: Новосибирск - Москва - Стокгольм - Цюрих



№	Имя	IP	AS	Время, мс
1	192.168.50.15	192.168.50.15		0,682
2	212.164.50.113	212.164.50.113	AS8691	1,119
3	87.226.133.75	87.226.133.75	AS12389	61,256
4	s-b9-link.ip.twelve99.net	213.155.129.46	AS1299	63,563
5	cms-publish.ethz.ch	129.132.19.216	AS559	93,374

wikimapia.org: Новосибирск - Москва - Лондон - Кассель (Германия)



№	Имя	IP	AS	Время, мс
1	192.168.50.15	192.168.50.15		0,934
2	212.164.50.113	212.164.50.113	AS8691	1,124
3	188.128.106.48	188.128.106.48	AS12389	1,261
4	telia-gw.fnt.cw.net	195.2.22.238	AS1273	82,621
5	www.wikimapia.org	88.99.95.180	AS24940	91,351

Код программы

```
#!/usr/bin/env python3
```

```
import os, sys, socket, struct, select, time, signal
```

```
timeout = 1000  
numPackets = 4  
packetSize = 64
```

```

maxSleep = 1000

ICMP_ECHO = 8 # echo request type
ICMP_MAX_RECV = 1024 # max size of incoming buffer

class Stats:
    packetsSent = 0
    packetsReceived = 0
    minTime = 10000
    maxTime = 0
    avgTime = 0
    totalTime = 0
    packetLoss = 1.0
    curIP = '0.0.0.0'

stats = Stats
timer = time.time

def ping(hostname):

    signal.signal(signal.SIGINT, exitHandler) # Ctrl+C handler
    seqNum = 0

    try:
        targetIP = socket.gethostbyname(hostname)
        print('ping.py: %s (%s)' % (hostname, targetIP))
    except socket.gaierror as e:
        print('ping.py: Unable to get hostname %s (%s)' % (hostname, e.args[1]))
        sys.exit(0)

    stats.curIP = targetIP

    for i in range(numPackets):
        delay = getDelay(stats, targetIP, hostname, timeout, seqNum, packetSize)
        seqNum += 1

        if delay == None:
            delay = 0

        if (maxSleep > delay):
            time.sleep((maxSleep - delay) / 1000) # ping intervals

    printStats(stats)

def getDelay(stats, targetIP, hostname, timeout, seqNum, packetSize):

    delay = None

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_RAW,
socket.getprotobyname('ICMP'))
    except socket.error as e:
        print('failed. (Socket error: "%s")' % e.args[1])
        raise

```

```

curProcID = os.getpid() & 0xFFFF

sentTime = pingSend(sock, targetIP, curProcID, seqNum, packetSize)
if sentTime == None:
    sock.close()
    return delay

stats.packetsSent += 1

recvTime, dataSize, iphSrcIP, icmpSeqNumber, iphTTL = pingRcv(sock,
curProcID, timeout)
sock.close()

if recvTime:
    delay = (recvTime - sentTime) * 1000
    print('%d bytes from %s: icmp_seq=%d ttl=%d time=%d ms' % (dataSize,
socket.inet_ntoa(struct.pack('!I', iphSrcIP)), icmpSeqNumber, iphTTL, delay))
    stats.packetsReceived += 1
    stats.totalTime += delay
    if stats.minTime > delay:
        stats.minTime = delay
    if stats.maxTime < delay:
        stats.maxTime = delay
else:
    delay = None
    print('Request timed out.')

return delay

def pingSend(sock, targetIP, ID, seqNum, packetSize):

    padBytes = []
    startVal = 0x42 # 66
    checksum = 0

    # ICMP header
    type, code (8) (16) (16) (16)
    header = struct.pack('!BBHH', ICMP_ECHO, 0, checksum, ID, seqNum)

    # (packetSize - 8) removes header size from packet size
    for i in range(startVal, startVal + (packetSize - 8)):
        padBytes += [(i & 0xff)] # keep chars in the 0-255 range

    data = bytearray(padBytes) # convert list to byte array
    checksum = getChecksum(header + data)
    header = struct.pack('!BBHH', ICMP_ECHO, 0, checksum, ID, seqNum)
    packet = header + data # packet to send

    sendTime = timer()

    try:
        sock.sendto(packet, (targetIP, 1)) # the port number is irrelevant
    except socket.error as e:
        print('General failure (%s)' % (e.args[1]))

    return sendTime

```



```

def pingRcv(sock, ID, timeout):

    timeLeft = timeout / 1000

    while True: # Loop while waiting for packet or timeout
        startMonitoring = timer()
        whatReady = select.select([sock], [], [], timeLeft) # monitoring socket
        monitoringTime = (timer() - startMonitoring)
        if whatReady[0] == []: # timeout
            return None, 0, 0, 0, 0

        timeReceived = timer()

        recPacket, addr = sock.recvfrom(ICMP_MAX_RECV)

        ipHeader = recPacket[:20]
        iphVersion, iphTypeOfSvc, iphLength, iphID, iphFlags, iphTTL,
        iphProtocol, iphChecksum, iphSrcIP, iphDestIP = struct.unpack('!BBHHHBBHII',
        ipHeader)

        icmpHeader = recPacket[20:28]
        icmpType, icmpCode, icmpChecksum, icmpPacketID, icmpSeqNumber =
        struct.unpack('!BBHHH', icmpHeader)

        if icmpPacketID == ID: # if our packet
            dataSize = len(recPacket) - 28
            return timeReceived, (dataSize + 8), iphSrcIP, icmpSeqNumber, iphTTL

        timeLeft = timeLeft - monitoringTime
        if timeLeft <= 0:
            return None, 0, 0, 0, 0

def getChecksum(sourceString):

    sum, numPackets, loByte, hiByte = 0, 0, 0, 0
    numPacketsTo = (int(len(sourceString) / 2)) * 2

    # handle bytes in pairs (decoding as short ints)
    while numPackets < numPacketsTo:
        loByte, hiByte = sourceString[numPackets], sourceString[numPackets + 1]
        sum = sum + (hiByte * 256 + loByte)
        numPackets += 2

    # handle last byte (if the number of bytes is odd)
    if numPacketsTo < len(sourceString): # check for odd length
        loByte = sourceString[len(sourceString)-1]
        sum += loByte

    sum = (sum >> 16) + (sum & 0xffff) # add high 16 bits to low 16 bits
    sum += (sum >> 16) # add carry from above (if any)
    answer = ~sum & 0xffff # invert and truncate to 16 bits
    answer = socket.htons(answer)

    return answer

```

```

def printStats(stats):

    print('\n--- %s ping statistics ---' % (stats.curIP))

    if stats.packetsSent > 0:
        stats.packetLoss = 100.0 * (stats.packetsSent - stats.packetsReceived) /
stats.packetsSent

    print('%d packets transmitted, %d packets received, %0.1f%% packet loss' %
(stats.packetsSent, stats.packetsReceived, stats.packetLoss))

    if stats.packetsReceived > 0:
        print('min/avg/max = %d/%0.1f/%d ms' % (stats.minTime, stats.totalTime /
stats.packetsReceived, stats.maxTime))

def exitHandler(frame, signal):

    printStats(stats)
    print('\n(Terminated)')

    sys.exit(0)

if __name__ == '__main__':
    ping(sys.argv[1])

```

Результаты работы программы

Сравнение разработанной программы с утилитой ping:

```

~/university-tasks/Computer Networks > master ± ping google.com
PING google.com (172.217.21.174) 56(84) bytes of data.
64 bytes from google.com (172.217.21.174): icmp_seq=1 ttl=57 time=64.8 ms
64 bytes from google.com (172.217.21.174): icmp_seq=2 ttl=57 time=64.6 ms
64 bytes from google.com (172.217.21.174): icmp_seq=3 ttl=57 time=65.6 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 64.601/65.019/65.625/0.438 ms
~/university-tasks/Computer Networks > master ± sudo python ping.py google.com
ping.py: google.com (172.217.21.174)
64 bytes from 172.217.21.174: icmp_seq=0 ttl=57 time=66 ms
64 bytes from 172.217.21.174: icmp_seq=1 ttl=57 time=65 ms
64 bytes from 172.217.21.174: icmp_seq=2 ttl=57 time=64 ms
64 bytes from 172.217.21.174: icmp_seq=3 ttl=57 time=63 ms

--- 172.217.21.174 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
min/avg/max = 63/65.0/66 ms

```

Проверка обработки программой сигнала Ctrl+C (остановка процесса):


```
~/university-tasks/Computer Networks > master ± sudo python ping.py google.com
ping.py: google.com (172.217.21.174)
64 bytes from 172.217.21.174: icmp_seq=0 ttl=57 time=66 ms
64 bytes from 172.217.21.174: icmp_seq=1 ttl=57 time=64 ms
^C
--- 172.217.21.174 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
min/avg/max = 64/65.7/66 ms
```

(Terminated)

Проверка обработки программой несуществующего доменного имени:

```
~/university-tasks/Computer Networks > master ± sudo python ping.py coogle.gom
ping.py: Unable to get hostname coogle.gom (Name or service not known)
```