**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**



ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ

**Лабораторная работа №5**

по дисциплине: Компьютерные сети

тема: «Протоколы ARP/RARP»

Выполнил: ст. группы ПВ-223

Пахомов Владислав Андреевич

Проверили:

Рубцов Константин Анатольевич

Белгород 2025 г.

**Лабораторная работа №5  
Программирование протоколов TCP/UDP с использованием библиотеки Winsock  
Вариант 6**

**Цель работы:** изучить протоколы ARP/RARP.

**Краткие теоретические сведения**

ARP (Address Resolution Protocol - протокол определения адреса) - протокол канального уровня, предназначенный для определения MACадреса (адреса канального уровня) по известному IP-адресу (адресу сетевого уровня). Наибольшее распространение этот протокол получил благодаря распространению сетей IP, построенных поверх Ethernet, поскольку практически в 100 % случаев при таком сочетании используется протокол ARP.

Протокол ARP работает различным образом в зависимости от того, какой протокол канального уровня работает в данной сети - протокол локальной сети (Ethernet, Token Ring, FDDI) с возможностью широковещательного доступа одновременно ко всем узлам сети, или же протокол глобальной сети (X.25, frame relay), как правило не поддерживающий широковещательный доступ.

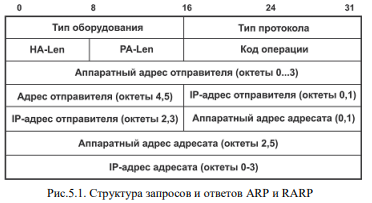
Функциональность протокола ARP сводится к решению двух задач. Одна часть протокола определяет физические адреса при посылке дейтаграммы, другая отвечает на запросы устройств в сети. Протокол ARP предполагает, что каждое устройство «знает» как свой IP -адрес, так и свой физический адрес.

Для того чтобы уменьшить количество посылаемых запросов ARP, каждое устройство в сети, использующее протокол ARP , должно иметь специальную буферную память. В ней хранятся пары адресов (IP-адрес, физический адрес) устройств в сети. Всякий раз, когда устройство получает ARP-ответ, оно сохраняет в буферной памяти соответствующую пару. Если адрес есть в списке пар, то нет необходимости посылать ARP-запрос. Эта буферная память называется ARP-таблицей.

В ARP-таблице могут содержаться как статические, так и динамические записи. Динамические записи добавляются и удаляются автоматически, статические заносятся вручную. Так как большинство устройств в сети поддерживает динамическое разрешение адресов, то администратору, как правило, нет необходимости собственноручно указывать записи протокола ARP в таблице адресов. Кроме того, ARP-таблица всегда содержит запись с физическим широковещательным адресом (OxFFFFFFFFFFFF) для локальной сети. Эта запись позволяет устройству принимать широковещательные ARP-запросы. Каждая запись в ARP-таблице имеет свое время жизни, например для операционной системы Microsoft Windows 2000 оно составляет 10 минут. При добавлении записи для нее активируется таймер. Если запись не востребована в первые две минуты, она удаляется. Если используется — будет существовать на протяжении 10 минут. В некоторых реализациях протокола ARP новый таймер устанавливается после каждого обращения к записи в ARP -таблице.

Сообщения протокола ARP при передаче по сети инкапсулируются в поле данных кадра. Они не содержат IP-заголовка. В отличие от сообщений большинства протоколов, сообщения ARP не имеют фиксированного формата заголовка. Это объясняется тем, что протокол был разработан таким образом, чтобы он был применим для разрешения адресов в различных сетях. Фактически протокол способен работать с произвольными физическими адресами и сетевыми протоколами.

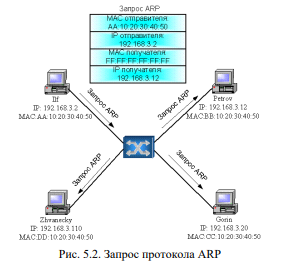
Узел, которому нужно выполнить отображение IP-адреса на локальный адрес, формирует ARP запрос, вкладывает его в кадр протокола канального уровня, указывая в нем известный IP-адрес, и рассылает запрос широковещательно. Все узлы локальной сети получают ARP запрос и сравнивают указанный там IP-адрес с собственным. В случае их совпадения узел формирует ARP-ответ, в котором указывает свой IP-адрес и свой локальный адрес и отправляет его уже направленно, так как в ARP запросе отправитель указывает свой локальный адрес. ARP-запросы и ответы используют один и тот же формат пакета. Так как локальные адреса могут в различных типах сетей иметь различную длину, то формат пакета протокола ARP зависит от типа сети. На рис. 5.1 показана структура запросов и ответов ARP и RARP.



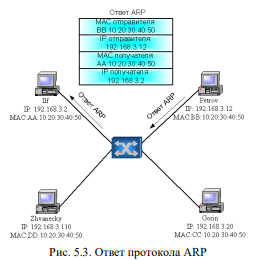
В поле типа оборудования для сетей Ethernet указывается значение 1. Поле типа протокола позволяет использовать пакеты ARP не только для протокола IP, но и для других сетевых протоколов. Для IP значение этого поля равно 0800 16. Длина локального адреса для протокола Ethernet равна 6 байтам, а длина IP-адреса - 4 байтам. В поле операции для ARP запросов указывается значение 1 для протокола ARP и 2 для протокола RARP.

Узел, отправляющий ARP-запрос, заполняет в пакете все поля, кроме поля искомого локального адреса (для RARP-запроса не указывается искомый IP-адрес). Значение этого поля заполняется узлом, опознавшим свой IP-адрес.

Рис. 5.2 и 5.3 показывают принцип работы протокола ARP. Анализируя структуру запроса, можно увидеть, что компьютер с адресом 192.168.3.2 делает попытку узнать МАС-адрес компьютера с IP -адресом 192.168.3.12. Для этого он посылает широковещательный запрос, содержащий IP-адрес, с МАС-адресом, установленным в FF:FF:FF:FF:FF:FF.



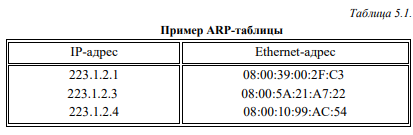
Когда компьютер с адресом 192.168.3.12 получает этот широковещательный запрос, он анализирует IP -адрес, для которого выполняется разрешение. Определив, что его адрес совпадает с искомым, он формирует ответ протокола ARP , где указывает свой МАС-адрес (рис. 5.3).



Ответ посылается уже не широковещательно - отправитель знает МАС-адрес инициатора запроса и поэтому передает пакет целенаправленно.

В глобальных сетях администратору сети чаще всего приходится вручную формировать ARP-таблицы, в которых он задает, например, соответствие IP-адреса адресу узла сети X.25, который имеет смысл локального адреса. В последнее время наметилась тенденция автоматизации работы протокола ARP и в глобальных сетях. Для этой цели среди всех маршрутизаторов, подключенных к какой-либо глобальной сети, выделяется специальный маршрутизатор, который ведет ARP-таблицу для всех остальных узлов и маршрутизаторов этой сети. При таком централизованном подходе для всех узлов и маршрутизаторов вручную нужно задать только IP-адрес и локальный адрес выделенного маршрутизатора. Затем каждый узел и маршрутизатор регистрирует свои адреса в выделенном маршрутизаторе, а при необходимости установления соответствия между IP-адресом и локальным адресом узел обращается к выделенному маршрутизатору с запросом и автоматически получает ответ без участия администратора.

Преобразование адресов выполняется путем поиска в таблице. Эта таблица, называемая ARP-таблицей, хранится в памяти и содержит 49 строки для каждого узла сети. В двух столбцах содержатся IP- и Ethernet-адреса. Если требуется преобразовать IP-адрес в Ethernet-адрес, то ищется запись с соответствующим IP-адресом (табл. 5.1.).



Принято все байты 4-байтного IP-адреса записывать десятичными числами, разделенными точками. При записи 6-байтного Ethernetадреса каждый байт указывается в 16-ричной системе и отделяется двоеточием.

ARP-таблица необходима потому, что IP-адреса и Ethernet-адреса выбираются независимо, и нет какого-либо алгоритма для преобразования одного в другой. IP-адрес выбирает менеджер сети с учетом положения машины в сети internet. Если машину перемещают в другую часть сети internet, то ее IP-адрес должен быть изменен. Ethernet-адрес выбирает производитель сетевого интерфейсного оборудования из выделенного для него по лицензии адресного пространства. Когда у машины заменяется плата сетевого адаптера, то меняется и ее Ethernet-адрес.

В ходе обычной работы сетевая программа, например TELNET, отправляет прикладное сообщение, пользуясь транспортными услугами TCP. Модуль TCP посылает соответствующее транспортное сообщение через модуль IP. В результате составляется IP-пакет, который должен быть передан драйверу Ethernet. IP-адрес места назначения известен прикладной программе, модулю TCP и модулю IP. Необходимо на его основе найти Ethernet-адрес места назначения. Для определения искомого Ethernet-адреса используется ARP-таблица.

ARP-таблица заполняется автоматически модулем ARP, по мере необходимости. Когда с помощью существующей ARP-таблицы не удается преобразовать IP-адрес, то происходит следующее:

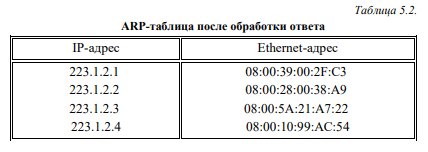
1. По сети передается широковещательный ARP-запрос.

2. Исходящий IP-пакет ставится в очередь.

Каждый сетевой адаптер принимает широковещательные передачи. Все драйверы Ethernet проверяют поле типа в принятом Ethernet-кадре и передают ARP-пакеты модулю ARP.

Каждый модуль ARP проверяет поле искомого IP-адреса в полученном ARP-пакете и, если адрес совпадает с его собственным IPадресом, то посылает ответ прямо по Ethernet-адресу отправителя запроса.

Этот ответ получает машина, сделавшая ARP-запрос. Драйвер этой машины проверяет поле типа в Ethernet-кадре и передает ARPпакет модулю ARP. Модуль ARP анализирует ARP-пакет и добавляет запись в свою ARP-таблицу (см. табл. 5.2.).



Полностью порядок преобразования адресов выглядит так:

1. По сети передается широковещательный ARP-запрос.

2. Исходящий IP-пакет ставится в очередь.

3. Возвращается ARP-ответ, содержащий информацию о соответствии IP- и Ethernet-адресов. Эта информация заносится в ARP-таблицу.

4. Для преобразования IP-адреса в Ethernet-адрес у IP-пакета, постав ленного в очередь, используется ARP-таблица.

5. Ethernet-кадр передается по сети Ethernet.

Т.е., если с помощью ARP-таблицы не удается сразу осуществить преобразование адресов, то IP-пакет ставится в очередь, а необходимая для преобразования информация получается с помощью запросов и ответов протокола ARP, после чего IP-пакет передается по назначению.

Если в сети нет машины с искомым IP-адресом, то ARP-ответа не будет и не будет записи в ARP-таблице. Протокол IP будет уничтожать IP-пакеты, направляемые по этому адресу. Протоколы верхнего уровня не могут отличить случай повреждения сети Ethernet от случая отсутствия машины с искомым IP-адресом.

Некоторые реализации IP и ARP не ставят в очередь IP-пакеты на то время, пока они ждут ARP-ответов. Вместо этого IP-пакет просто уничтожается, а его восстановление возлагается на модуль TCP или прикладной процесс, работающий через UDP. Такое восстановление выполняется с помощью таймаутов и повторных передач. Повторная передача сообщения проходит успешно, так как первая попытка уже вызвала заполнение ARP-таблицы.

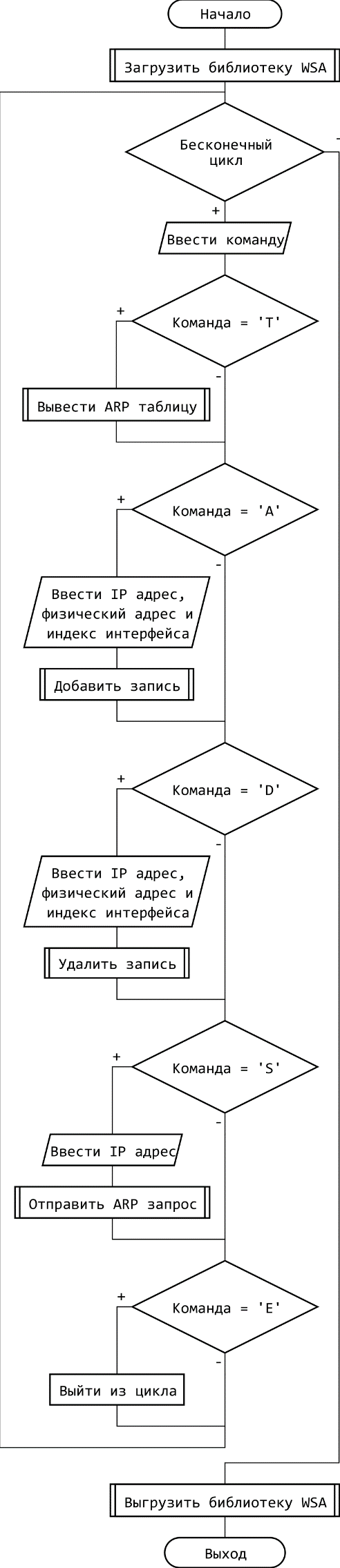
Следует отметить, что каждая машина имеет отдельную ARP-таблицу для каждого своего сетевого интерфейса.

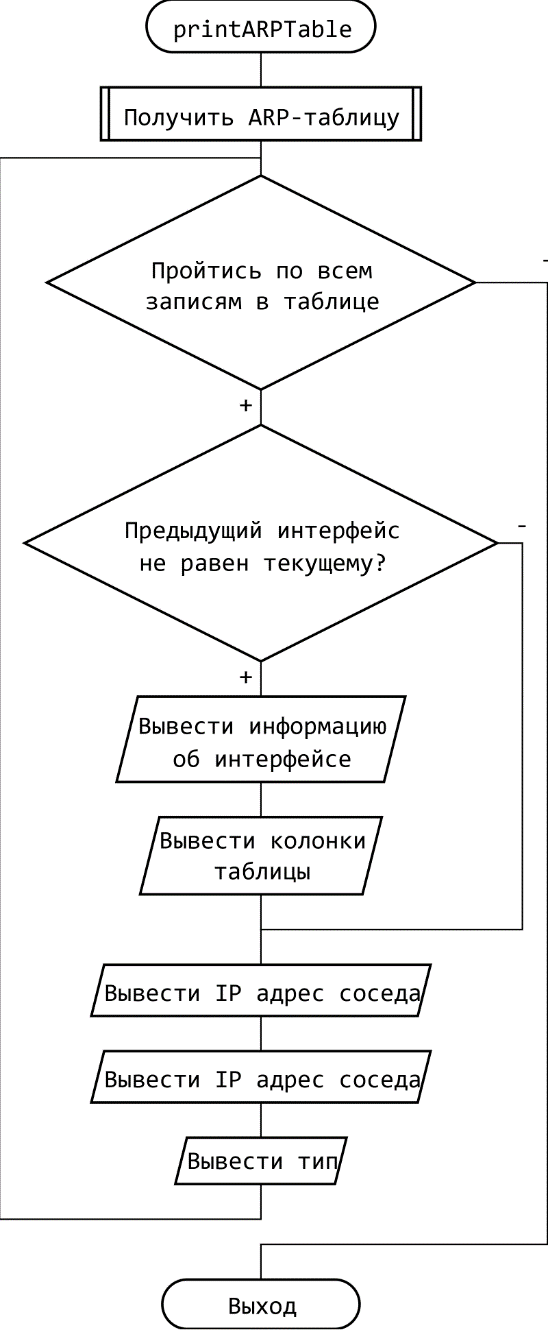
Протокол RARP - это протокол, решающий обратную задачу - нахождение IP-адреса по известному локальному адресу. Он называется реверсивный ARP - RARP (Reverse Address Resolution Protocol) и используется при старте бездисковых станций, не знающих в начальный момент своего IP-адреса, но знающих адрес своего сетевого адаптера. Reverse ARP (или обратное разрешение) работает аналогично протоколу ARP за исключением того, что в его задачи входит определение физического адреса по известному адресу сетевого уровня. Этот протокол требует наличия в сети сервера RARP , подключенного к тому же сегменту сети, что и интерфейс маршрутизатора. Наиболее часто протокол reverse ARP используется для запуска бездисковых рабочих станций.

**Используемые функции**

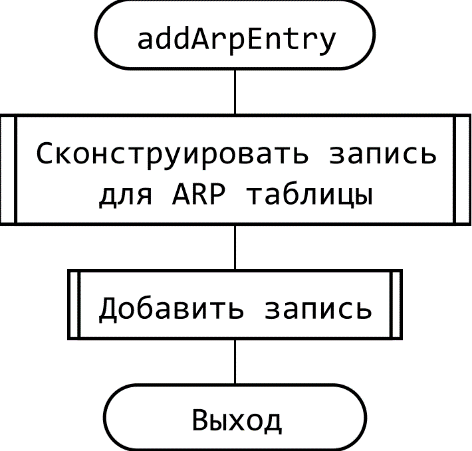
* **GetAdaptersAddresses –** возвращает информацию об интерфейсах текущего компьютера. Family – AF\_INET для данной лабораторной работы, Flags – флаги, Reserverd – неиспользуемое поле, AdapterAdresses – указатель на буфер с адресами, SizePointer – указатель на размер буфера. Если размер буфера недостаточно большой, в SizePointer пишется необходимое количество памяти.
* **GetIpNetTable2 –** возвращает ARP-таблицу, Family – AF\_INET для данной лабораторной работы, Table – указатель на табллицу.
* **CreateIpNetEntry2 –** добавляет запись в ARP-таблицу, Row – адрес на добавляемый ряд
* **DeleteIpNetEntry2 –** удаляет запись из ARP-таблицы, Row – адрес на удаляемый ряд
* **SendARP –** отправляет ARP-запрос, DestIP – IP адрес который нужно найти, SrcIP – принимающий IP адрес, можно указать ADDR\_ANY, pMacAddr – указатель на MAC-адрес, результат работы ARP-запроса, PhyAddrLen – указатель на длину MAC-адреса

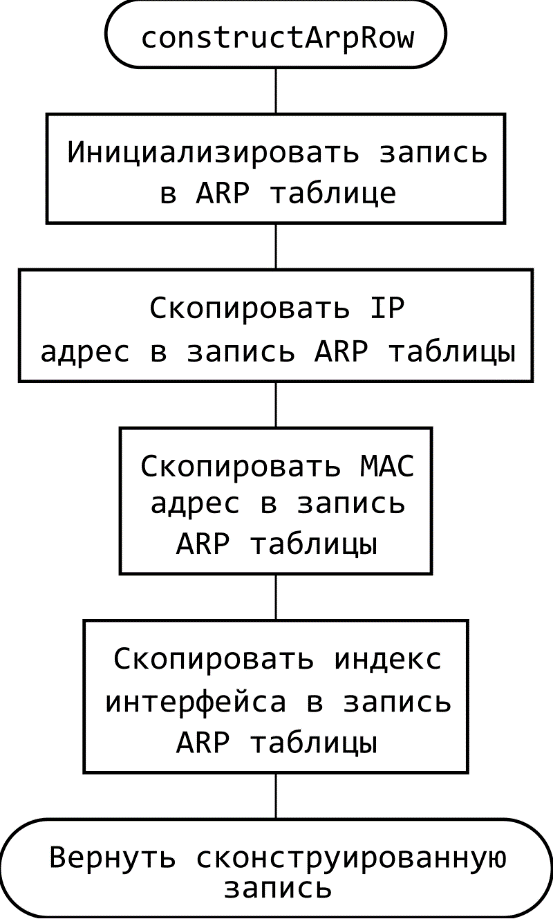
**Разработка программы. Блок-схемы программы.**

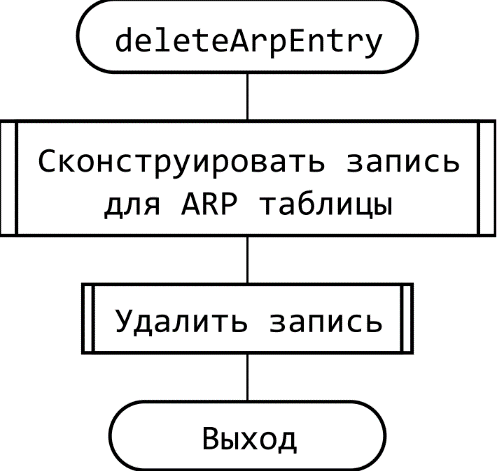
****

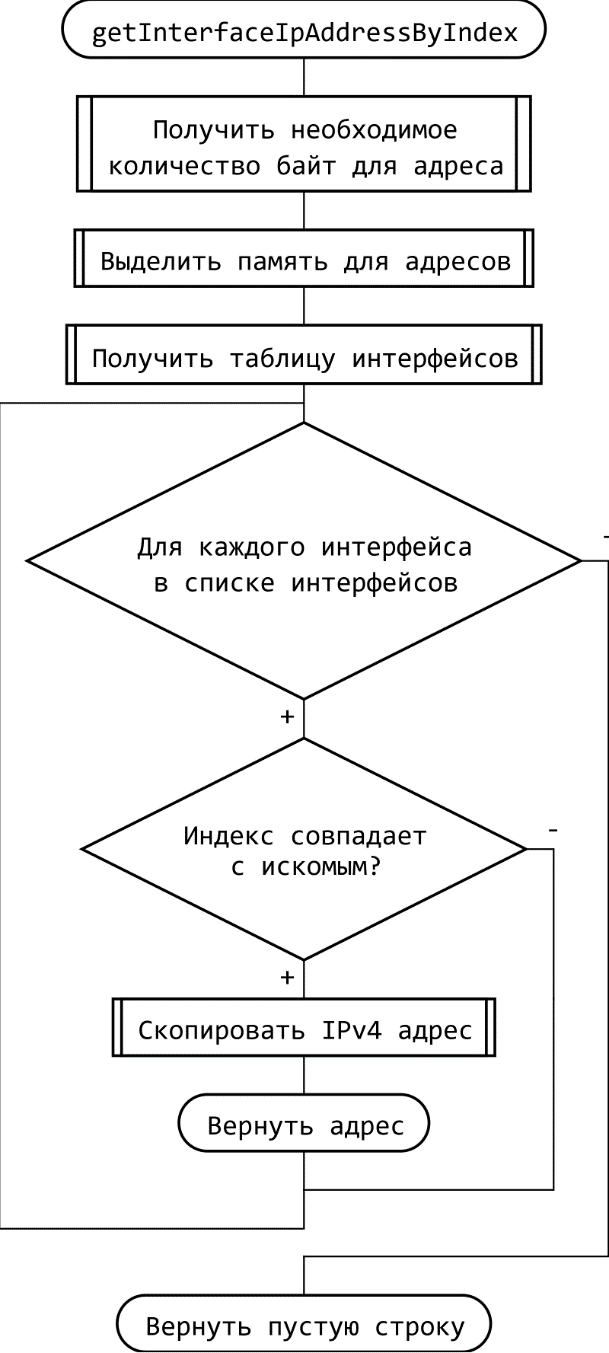
****

****

****

****

****

****

**Анализ функционирования программ**

* **Вывод**Программа выводит ARP-таблицу для текущего устройства

Интерфейс: 127.0.0.1 --- 0x1

Адрес в Интернете Физический адрес Тип

127.0.0.1 статический

224.0.0.2 статический

224.0.0.22 статический

224.250.207.56 статический

239.255.255.250 статический

Интерфейс: 169.254.248.173 --- 0x17

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.1.175 --- 0xa

Адрес в Интернете Физический адрес Тип

80.250.175.30 00:00:00:00:00:00 недопустимый

169.254.255.255 00:00:00:00:00:00 недопустимый

192.168.1.1 78:8c:b5:84:18:34 динамический

192.168.1.176 08:00:27:7b:2c:a3 устаревший

192.168.1.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 169.254.134.15 --- 0x13

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 169.254.38.233 --- 0xe

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

Интерфейс: 169.254.20.31 --- 0xd

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 172.24.16.1 --- 0x19

Адрес в Интернете Физический адрес Тип

172.24.31.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.56.1 --- 0x5

Адрес в Интернете Физический адрес Тип

192.168.56.1 00:00:00:00:00:00 недопустимый

192.168.56.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

* **Добавление записи**Запись успешно добавляется в ARP-таблицу, после добавления записи повторный вывод показывает новую запись (выделено красным цветом)

IP адрес: 192.168.1.242

MAC адрес: 08:00:27:5a:df:9e

Индекс интерфейса: 10

Интерфейс: 127.0.0.1 --- 0x1

Адрес в Интернете Физический адрес Тип

127.0.0.1 статический

224.0.0.2 статический

224.0.0.22 статический

224.250.207.56 статический

239.255.255.250 статический

Интерфейс: 169.254.248.173 --- 0x17

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.1.175 --- 0xa

Адрес в Интернете Физический адрес Тип

80.250.175.30 00:00:00:00:00:00 недопустимый

169.254.255.255 00:00:00:00:00:00 недопустимый

192.168.1.1 78:8c:b5:84:18:34 динамический

192.168.1.176 08:00:27:7b:2c:a3 устаревший

192.168.1.242 08:00:27:5a:df:9e статический

192.168.1.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 169.254.134.15 --- 0x13

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 169.254.38.233 --- 0xe

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

Интерфейс: 169.254.20.31 --- 0xd

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 172.24.16.1 --- 0x19

Адрес в Интернете Физический адрес Тип

172.24.31.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.56.1 --- 0x5

Адрес в Интернете Физический адрес Тип

192.168.56.1 00:00:00:00:00:00 недопустимый

192.168.56.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

* **Удаление записи**

Можем удалить добавленную только что запись, удаление также работает

IP адрес: 192.168.1.242

MAC адрес: 08:00:27:5a:df:9e

Индекс интерфейса: 10

Интерфейс: 127.0.0.1 --- 0x1

Адрес в Интернете Физический адрес Тип

127.0.0.1 статический

224.0.0.2 статический

224.0.0.22 статический

224.250.207.56 статический

239.255.255.250 статический

Интерфейс: 169.254.248.173 --- 0x17

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.1.175 --- 0xa

Адрес в Интернете Физический адрес Тип

80.250.175.30 00:00:00:00:00:00 недопустимый

169.254.255.255 00:00:00:00:00:00 недопустимый

192.168.1.1 78:8c:b5:84:18:34 динамический

192.168.1.176 08:00:27:7b:2c:a3 устаревший

192.168.1.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 169.254.134.15 --- 0x13

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 169.254.38.233 --- 0xe

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

Интерфейс: 169.254.20.31 --- 0xd

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 172.24.16.1 --- 0x19

Адрес в Интернете Физический адрес Тип

172.24.31.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.56.1 --- 0x5

Адрес в Интернете Физический адрес Тип

192.168.56.1 00:00:00:00:00:00 недопустимый

192.168.56.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

* **Отправка ARP**

Запрос также успешно отправляется, если IP адрес уже существует в сети. При этом запись автоматически добавляется после отправки запроса. При таком способе отправки ARP-запросе тип записи уже является динамическим, а не статическим.

IP адрес: 192.168.1.242

Полученный MAC-адрес: 08:00:27:5a:fd:9e

Интерфейс: 127.0.0.1 --- 0x1

Адрес в Интернете Физический адрес Тип

127.0.0.1 статический

224.0.0.2 статический

224.0.0.22 статический

224.250.207.56 статический

239.255.255.250 статический

Интерфейс: 169.254.248.173 --- 0x17

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.1.175 --- 0xa

Адрес в Интернете Физический адрес Тип

80.250.175.30 00:00:00:00:00:00 недопустимый

169.254.255.255 00:00:00:00:00:00 недопустимый

192.168.1.1 78:8c:b5:84:18:34 динамический

192.168.1.176 08:00:27:7b:2c:a3 устаревший

192.168.1.242 08:00:27:5a:fd:9e динамический

192.168.1.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 169.254.134.15 --- 0x13

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 169.254.38.233 --- 0xe

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

Интерфейс: 169.254.20.31 --- 0xd

Адрес в Интернете Физический адрес Тип

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.250.207.56 01:00:5e:7a:cf:38 статический

Интерфейс: 172.24.16.1 --- 0x19

Адрес в Интернете Физический адрес Тип

172.24.31.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

255.255.255.255 ff:ff:ff:ff:ff:ff статический

Интерфейс: 192.168.56.1 --- 0x5

Адрес в Интернете Физический адрес Тип

192.168.56.1 00:00:00:00:00:00 недопустимый

192.168.56.255 ff:ff:ff:ff:ff:ff статический

224.0.0.2 01:00:5e:00:00:02 статический

224.0.0.22 01:00:5e:00:00:16 статический

224.0.0.251 01:00:5e:00:00:fb статический

224.0.0.252 01:00:5e:00:00:fc статический

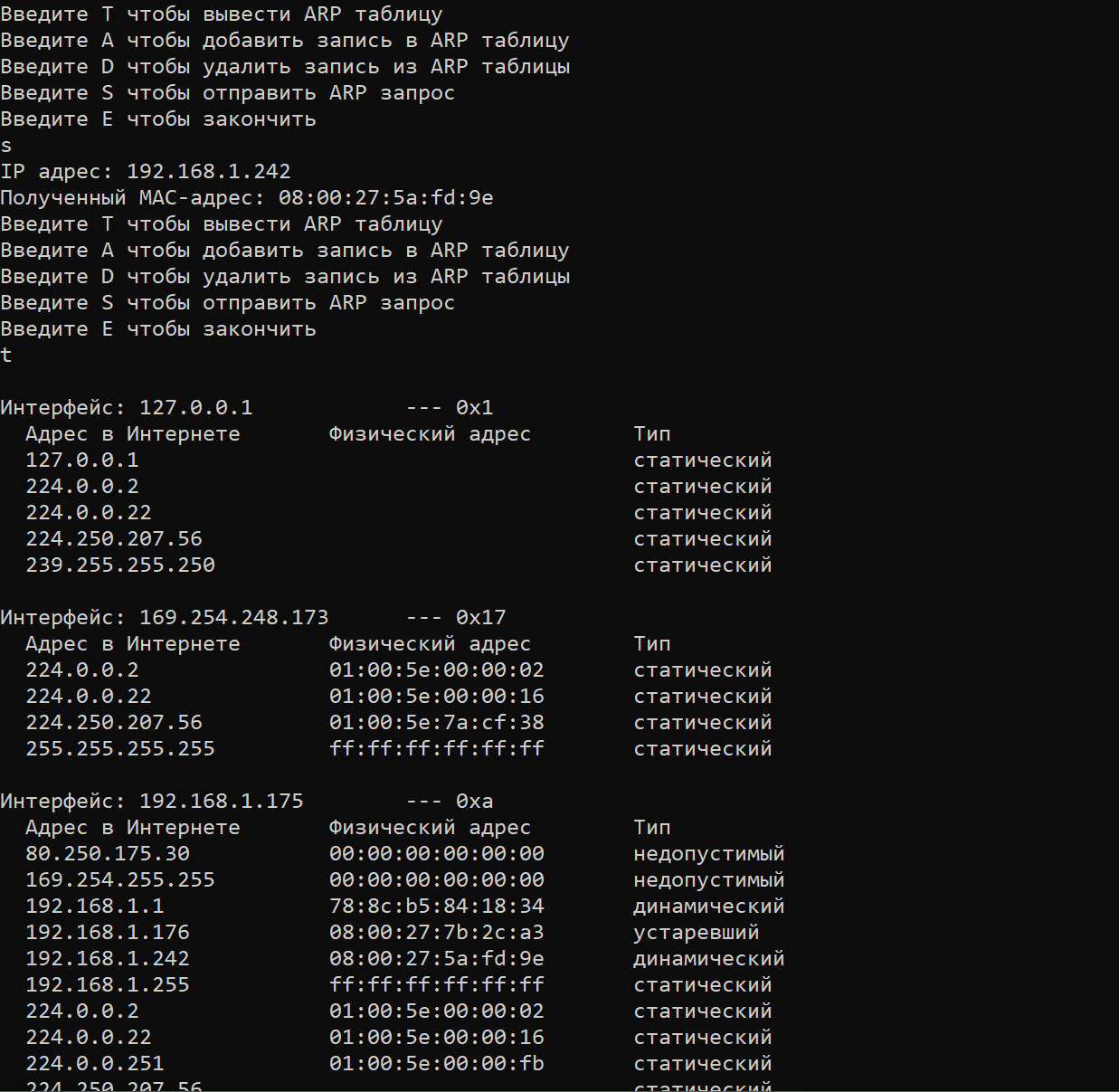
224.250.207.56 01:00:5e:7a:cf:38 статический

239.255.255.250 01:00:5e:7f:ff:fa статический

**Вывод:** в ходе лабораторной изучили протоколы ARP/RARP, написали программу позволяющую редактировать и просматривать ARP-таблицу и отправлять ARP-запросы.

**Текст программ. Скриншоты программ.**

Ссылка на репозиторий с кодом: <https://github.com/IAmProgrammist/comp_net/tree/lab5>



#include <iostream>

#include <algorithm>

#include <webstur/arp.h>

#include <webstur/iserver.h>

int main() {

    setlocale(LC\_ALL, "Russian");

    try {

        IServer::init();

        std::string input;

        // Считытваем команду пользователя

        while (true)

        {

            std::cout << "Введите T чтобы вывести ARP таблицу\n"

                      << "Введите A чтобы добавить запись в ARP таблицу\n"

                      << "Введите D чтобы удалить запись из ARP таблицы\n"

                      << "Введите S чтобы отправить ARP запрос\n"

                      << "Введите E чтобы закончить" << std::endl;

            std::cin >> input;

            std::transform(input.begin(), input.end(), input.begin(), toupper);

            if (input == "T") {

                // Вывести ARP таблицу

                ARPHelper::printARPTable(std::cout);

            }

            else if (input == "A") {

                // Ввести IP адрес, физический адрес и индекс интерфейса

                std::string ip\_address, physical\_address;

                int interface\_index;

                std::cout << "IP адрес: ";

                std::cout.flush();

                std::cin >> ip\_address;

                std::cout << "MAC адрес: ";

                std::cout.flush();

                std::cin >> physical\_address;

                std::cout << "Индекс интерфейса: ";

                std::cout.flush();

                std::cin >> interface\_index;

                // Добавить запись

                ARPHelper::addArpEntry(ip\_address, physical\_address, interface\_index);

            }

            else if (input == "D") {

                // Ввести IP адрес, физический адрес и индекс интерфейса

                std::string ip\_address, physical\_address;

                int interface\_index;

                std::cout << "IP адрес: ";

                std::cout.flush();

                std::cin >> ip\_address;

                std::cout << "MAC адрес: ";

                std::cout.flush();

                std::cin >> physical\_address;

                std::cout << "Индекс интерфейса: ";

                std::cout.flush();

                std::cin >> interface\_index;

                // Удалить запись

                ARPHelper::deleteArpEntry(ip\_address, physical\_address, interface\_index);

            }

            else if (input == "S") {

                // Ввести IP адрес

                std::string ip\_address;

                std::cout << "IP адрес: ";

                std::cout.flush();

                std::cin >> ip\_address;

                // Отправить ARP запрос

                std::cout << "Полученный MAC-адрес: " << ARPHelper::sendARP(ip\_address) << std::endl;

            }

            else if (input == "E") {

                // Выход из цикла

                break;

            }

        }

    }

    catch (const std::runtime\_error& error) {

        std::cerr << "Failed while running server. Caused by: '" << error.what() << "'" << std::endl;

        return -1;

    }

    // Выгрузка библиотеки WSA

    IServer::detach();

    return 0;

}

#pragma once

#include <ostream>

#include <webstur/utils.h>

class DLLEXPORT ARPHelper {

public:

    // Выводит таблицу ARP в указанный ostream

    static std::ostream& printARPTable(std::ostream& out);

    // Добавляет запись в ARP таблицу, где

    //   ip               адрес, строка в формате 255.255.255.255

    //   address          физический адрес, строка в формате ff:ff:ff:ff:ff:ff

    //   interface\_index  индекс интерфейса

    static void addArpEntry(std::string ip, std::string address, int interface\_index);

    // Удаляет запись из ARP таблицы, где

    //   ip               адрес, строка в формате 255.255.255.255

    //   address          физический адрес, строка в формате ff:ff:ff:ff:ff:ff

    //   interface\_index  индекс интерфейса

    static void deleteArpEntry(std::string ip, std::string address, int interface\_index);

    // Отправляет ARP запрос, где

    //   find             адрес, который нужно найти

    static std::string sendARP(std::string find);

    // Форматирует физический адрес

    static std::string prettyPrintPhysicalAddress(unsigned char\* addr, int addr\_size);

    // Возвращает IP адрес интерфейса по его индексу

    static std::string getInterfaceIpAddressByIndex(int dw\_index);

};

#include "pch.h"

#include <ws2ipdef.h>

#include <winsock2.h>

#include <iphlpapi.h>

#include <ws2tcpip.h>

#include <iostream>

#include <iomanip>

#include <sstream>

#include <webstur/arp.h>

std::string ARPHelper::prettyPrintPhysicalAddress(unsigned char\* addr, int addr\_size) {

    std::ostringstream result;

    for (int i = 0; i < addr\_size; i++) {

        if (i == addr\_size - 1) {

            result << std::setfill('0') << std::setw(2) << std::hex << (unsigned int) addr[i];

        } else {

            result << std::setfill('0') << std::setw(2) << std::hex << (unsigned int) addr[i] << ":";

        }

    }

    return result.str();

}

std::string ARPHelper::getInterfaceIpAddressByIndex(int dwIndex) {

    PIP\_ADAPTER\_ADDRESSES p\_addresses = NULL;

    unsigned long out\_buf\_len = 0;

    int dw\_ret\_val = 0;

    auto flags = GAA\_FLAG\_SKIP\_ANYCAST & GAA\_FLAG\_SKIP\_MULTICAST & GAA\_FLAG\_SKIP\_DNS\_SERVER;

    // Получить необходимое место для адреса

    dw\_ret\_val = GetAdaptersAddresses(AF\_INET,

        flags,

        NULL,

        NULL,

        &out\_buf\_len);

    // Выделить память для адресов

    p\_addresses = (PIP\_ADAPTER\_ADDRESSES)malloc(out\_buf\_len);

    // Получаем таблицу интерфейсов

    dw\_ret\_val = GetAdaptersAddresses(AF\_INET,

        flags,

        NULL,

        p\_addresses,

        &out\_buf\_len);

    if (dw\_ret\_val != ERROR\_SUCCESS) {

        free(p\_addresses);

        throw std::runtime\_error("Unable to get adapters table: " + std::to\_string(dw\_ret\_val));

    }

    // Пройтись по всем интерфейсам

    PIP\_ADAPTER\_ADDRESSES p\_curr\_addresses = p\_addresses;

    while (p\_curr\_addresses) {

        // Если индекс совпадает с искомым

        if (p\_curr\_addresses->IfIndex == dwIndex) {

            PIP\_ADAPTER\_UNICAST\_ADDRESS p\_unicast = p\_curr\_addresses->FirstUnicastAddress;

            int i = 0;

            while (p\_unicast != NULL) {

                char result[20] = {};

                if (p\_unicast->Address.lpSockaddr->sa\_family == AF\_INET) {

                    // Скопировать IPv4 адрес и вернуть его

                    sockaddr\_in\* sa\_in = (sockaddr\_in\*)p\_unicast->Address.lpSockaddr;

                    inet\_ntop(AF\_INET, &(sa\_in->sin\_addr), result, sizeof(result));

                    free(p\_addresses);

                    return std::string(result, result + sizeof(result));

                }

                p\_unicast = p\_unicast->Next;

            }

            break;

        }

        p\_curr\_addresses = p\_curr\_addresses->Next;

    }

    free(p\_addresses);

    return "";

}

std::ostream& ARPHelper::printARPTable(std::ostream& out) {

    // Получить ARP-таблицу

    PMIB\_IPNET\_TABLE2 pipTable = NULL;

    auto status = GetIpNetTable2(AF\_INET, &pipTable);

    if (status != NO\_ERROR)

        throw std::runtime\_error("GetIpNetTable for IPv4 table returned error: " + std::to\_string(status));

    std::string last\_interface = "";

    // Пройтись по всем записям в таблице

    for (int i = 0; i < pipTable->NumEntries; i++) {

        std::string interface = getInterfaceIpAddressByIndex(pipTable->Table[i].InterfaceIndex);

        if (interface != last\_interface) {

            // Вывести информацию об интерфейсе

            out << "\nИнтерфейс: " << getInterfaceIpAddressByIndex(pipTable->Table[i].InterfaceIndex) <<

                " --- 0x" << std::hex << pipTable->Table[i].InterfaceIndex << "\n" << std::dec;

            // Вывести колонки

            out << "  " <<

                std::left << std::setfill(' ') << std::setw(24) << "Адрес в Интернете" <<

                std::left << std::setfill(' ') << std::setw(24) << "Физический адрес" <<

                std::left << std::setfill(' ') << std::setw(24) << "Тип" << std::endl;

            last\_interface = interface;

        }

        // Вывести IP адрес соседа

        char temp\_string[16] = {};

        inet\_ntop(AF\_INET, &pipTable->Table[i].Address.Ipv4.sin\_addr, temp\_string, sizeof(temp\_string));

        out << "  " << std::left << std::setfill(' ') << std::setw(24) << std::string(temp\_string, temp\_string + strlen(temp\_string));

        // Вывести MAC адрес соседа

        out << std::left << std::setfill(' ') << std::setw(24) << prettyPrintPhysicalAddress(pipTable->Table[i].PhysicalAddress, pipTable->Table[i].PhysicalAddressLength);

        // Вывести тип

        std::string type;

        switch (pipTable->Table[i].State) {

        case NlnsUnreachable:

            type = "недопустимый";

            break;

        case NlnsIncomplete:

            type = "незавершённый";

            break;

        case NlnsProbe:

            type = "исследуется";

            break;

        case NlnsDelay:

            type = "задержан";

            break;

        case NlnsStale:

            type = "устаревший";

            break;

        case NlnsReachable:

            type = "динамический";

            break;

        case NlnsPermanent:

            type = "статический";

            break;

        case NlnsMaximum:

            type = "максимальный";

            break;

        }

        out << std::left << std::setfill(' ') << std::setw(24) << type;

        out << "\n";

    }

    FreeMibTable(pipTable);

    out.flush();

    return out;

}

MIB\_IPNET\_ROW2 constructArpRow(std::string ip, std::string address, int interface\_index) {

    // Инициализировать запись в ARP таблице

    MIB\_IPNET\_ROW2 entry = {};

    // Скопировать IP адрес в запись ARP таблицы

    entry.Address.si\_family = AF\_INET;

    inet\_pton(AF\_INET, &ip[0], &entry.Address.Ipv4.sin\_addr);

    // Скопировать MAC адрес в запись ARP таблицы

    entry.PhysicalAddressLength = 6;

    sscanf\_s(&address[0], "%02X:%02X:%02X:%02X:%02X:%02X",

        &entry.PhysicalAddress[0],

        &entry.PhysicalAddress[1],

        &entry.PhysicalAddress[2],

        &entry.PhysicalAddress[3],

        &entry.PhysicalAddress[4],

        &entry.PhysicalAddress[5]);

    // Скопировать индекс интерфейса в запись ARP таблицы

    entry.InterfaceIndex = interface\_index;

    return entry;

}

void ARPHelper::addArpEntry(std::string ip, std::string address, int interface\_index) {

    // Сконструировать запись для ARP таблицы

    auto new\_entry = constructArpRow(ip, address, interface\_index);

    new\_entry.State = NlnsPermanent;

    // Добавить запись

    int return\_code;

    if ((return\_code = CreateIpNetEntry2(&new\_entry)) != NO\_ERROR)

        throw std::runtime\_error("Unable to add entry with error code " + std::to\_string(return\_code));

}

void ARPHelper::deleteArpEntry(std::string ip, std::string address, int interface\_index) {

    // Сконструировать запись для ARP таблицы

    auto new\_entry = constructArpRow(ip, address, interface\_index);

    // Удалить запись

    int return\_code;

    if ((return\_code = DeleteIpNetEntry2(&new\_entry)) != NO\_ERROR)

        throw std::runtime\_error("Unable to add entry with error code " + std::to\_string(return\_code));

}

std::string ARPHelper::sendARP(std::string find) {

    unsigned long size = sizeof(unsigned long) \* 2;

    unsigned char mac[sizeof(unsigned long) \* 2];

    memset(mac, 0xff, sizeof(mac));

    IPAddr ip;

    inet\_pton(AF\_INET, &find[0], &ip);

    // Отправить ARP запрос

    int return\_code;

    if ((return\_code = SendARP(ip, ADDR\_ANY, mac, &size)) != NO\_ERROR)

        throw std::runtime\_error("Unable to add entry with error code " + std::to\_string(return\_code));

    // Преобразовать адрес в строку

    return ARPHelper::prettyPrintPhysicalAddress(mac, size);

}