**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

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

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

дисциплина: Сети ЭВМ и телекоммуникации

тема: «Протоколы ARP/RARP»

Выполнил: ст. группы ВТ-32

Воскобойников И. С.

Проверил: Федотов Е.А.

Белгород 2021

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

**Задание к работе**

1. Программно реализовать вывод ARP-таблицы.

2. Программно реализовать добавление записи в ARP-таблицу.

3. Программно реализовать удаление записи из ARP-таблицы.

4. Программно реализовать получение MAC-адреса по IP-адресу.

**Краткие теоретические сведения**

ARP (Address Resolution Protocol - протокол определения адреса) - протокол канального уровня, предназначенный для определения MAC-адреса (адреса канального уровня) по известному IP-адресу. 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. 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.

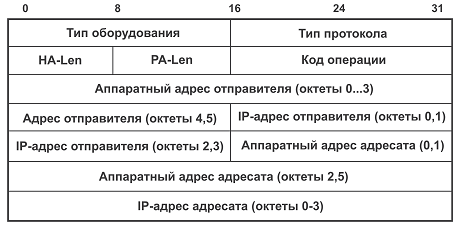


Рис.5.1. Структура запросов и ответов ARP и RARP

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.

Например, 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. 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 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 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.).

|  |  |
| --- | --- |
| IP-адрес | Ethernet-адрес |
| 223.1.2.1 223.1.2.3 223.1.2.4 | 08:00:39:00:2F:C3 08:00:5A:21:A7:22 08:00:10:99:AC:54 |

Таблица 5.1.

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.

ARP-таблица заполняется автоматически модулем ARP, по мере необходимости.

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.

Некоторые реализации IP и ARP не ставят в очередь IP-пакеты на то время, пока они ждут ARP-ответов.

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.

**Содержание файла lab5.cpp:**

#include <cstdlib>

#include <iostream>

#include <winsock.h>

#pragma comment(lib, "ws2\_32.lib")

#include <iphlpapi.h>

#pragma comment(lib, "iphlpapi.lib")

#pragma warning (disable: 4996)

#pragma warning (disable: 4477)

using namespace std;

DWORD OutputARPTable(){

PMIB\_IPNETTABLE IPARPTable = NULL;

DWORD ActualSize = 0;

GetIpNetTable(IPARPTable, &ActualSize, true);

IPARPTable = (PMIB\_IPNETTABLE)malloc(ActualSize);

if (GetIpNetTable(IPARPTable, &ActualSize, true) != NO\_ERROR){

cout << "Ошибка в получении таблицы\n" << endl;

if (IPARPTable)

free(IPARPTable);

return 1;

}

DWORD CurrentIndex;

char Type[256], Address[256];

PMIB\_IPADDRTABLE IPAddressTable = NULL;

ActualSize = 0;

GetIpAddrTable(IPAddressTable, &ActualSize, true);

IPAddressTable = (PMIB\_IPADDRTABLE)malloc(ActualSize);

GetIpAddrTable(IPAddressTable, &ActualSize, true);

CurrentIndex = -100;

for (DWORD i = 0; i < IPARPTable->dwNumEntries; i++){

if (IPARPTable->table[i].dwIndex != CurrentIndex){

CurrentIndex = IPARPTable->table[i].dwIndex;

IN\_ADDR InAddress;

for (DWORD j = 0; j < IPAddressTable->dwNumEntries; j++){

if (CurrentIndex != IPAddressTable->table[j].dwIndex)

continue;

InAddress.S\_un.S\_addr = IPAddressTable->table[j].dwAddr;

strcpy(Address, inet\_ntoa(InAddress));

}

printf("Интерфейс: %s --- 0x%X\n", Address, CurrentIndex);

cout << "Адрес в Интернете | Физический адрес | Тип" << endl;

}

switch (IPARPTable->table[i].dwType){

case 1:

strcpy(Type, "Другой");

break;

case 2:

strcpy(Type, "Неправильный");

break;

case 3:

strcpy(Type, "Динамический");

break;

case 4:

strcpy(Type, "Статический");

break;

default:

strcpy(Type, "");

}

IN\_ADDR InAddress;

InAddress.S\_un.S\_addr = IPARPTable->table[i].dwAddr;

printf("%-18s |", inet\_ntoa(InAddress));

printf(" %02X:%02X:%02X:%02X:%02X:%02X | %-11s\n", IPARPTable->table[i].bPhysAddr[0],

IPARPTable->table[i].bPhysAddr[1],

IPARPTable->table[i].bPhysAddr[2],

IPARPTable->table[i].bPhysAddr[3],

IPARPTable->table[i].bPhysAddr[4],

IPARPTable->table[i].bPhysAddr[5], Type);

}

free(IPARPTable);

cout << endl;

return 0;

}

DWORD AddIpNetEntry(){

char ArrayInetAddress[255], MacAddress[255], Interfase[255];

cout << "Введите интерфейс: "; cin >> Interfase;

cout << "Введите адрес в Интернете: "; cin >> ArrayInetAddress;

cout << "Введите физический адрес: "; cin >> MacAddress;

DWORD InetAddress = inet\_addr(ArrayInetAddress);

if (InetAddress == INADDR\_NONE){

cout << "Неправильный адрес в Интернете.\n" << endl;

return 1;

}

MIB\_IPNETROW ARPRow;

sscanf(Interfase, "%x", &(ARPRow.dwIndex));

ARPRow.dwPhysAddrLen = 6;

sscanf(MacAddress, "%hx:%hx:%hx:%hx:%hx:%hx",

&ARPRow.bPhysAddr[0],

&ARPRow.bPhysAddr[1],

&ARPRow.bPhysAddr[2],

&ARPRow.bPhysAddr[3],

&ARPRow.bPhysAddr[4],

&ARPRow.bPhysAddr[5]);

ARPRow.dwAddr = InetAddress;

ARPRow.dwType = MIB\_IPNET\_TYPE\_STATIC;

switch (CreateIpNetEntry(&ARPRow)){

case ERROR\_ACCESS\_DENIED:

cout << "Запись не добавлена. Доступ запрещен" << endl;

break;

case NO\_ERROR:

cout << "Запись добавлена" << endl;

break;

default:

cout << "Запись не добавлена" << endl;

}

return 0;

}

DWORD DropIpNetEntry(){

char ArrayInetAddress[255], Interfase[255];

cout << "Введите интерфейс: "; cin >> Interfase;

cout << "Введите адрес в Интернете: "; cin >> ArrayInetAddress;

DWORD InetAddress = inet\_addr(ArrayInetAddress);

if (InetAddress == INADDR\_NONE){

cout << "Неправильный адрес в Интернете" << endl;

return 1;

}

MIB\_IPNETROW ARPRow;

sscanf(Interfase, "%x", &(ARPRow.dwIndex));

ARPRow.dwAddr = InetAddress;

switch (DeleteIpNetEntry(&ARPRow)){

case ERROR\_ACCESS\_DENIED:

cout << "Запись не удалена. Доступ запрещен" << endl;

break;

case NO\_ERROR:

cout << "Запись удалена" << endl;

break;

default:

cout << "Запись не удалена" << endl;

}

return 0;

}

void IPtoMAC(){

DWORD ActualSize = 0;

PMIB\_IPNETTABLE IPAddressTable = NULL;

GetIpNetTable(IPAddressTable, &ActualSize, true);

IPAddressTable = (PMIB\_IPNETTABLE)malloc(ActualSize);

GetIpNetTable(IPAddressTable, &ActualSize, true);

char ArrayInetAddress[255];

cout << "Введите адрес в Интернете: "; cin >> ArrayInetAddress;

DWORD InetAddress = inet\_addr(ArrayInetAddress);

if (InetAddress == INADDR\_NONE){

cout << "Неправильный адрес в Интернете" << endl;

return;

}

bool SearchFlag = true;

for (DWORD i = 0; i < IPAddressTable->dwNumEntries; i++){

if (InetAddress == IPAddressTable->table[i].dwAddr){

printf("Физический адрес: %02X-%02X-%02X-%02X-%02X-%02X на интерфейсе с индексом %x\n",

IPAddressTable->table[i].bPhysAddr[0],

IPAddressTable->table[i].bPhysAddr[1],

IPAddressTable->table[i].bPhysAddr[2],

IPAddressTable->table[i].bPhysAddr[3],

IPAddressTable->table[i].bPhysAddr[4],

IPAddressTable->table[i].bPhysAddr[5], IPAddressTable->table[i].dwIndex);

SearchFlag = false;

}

}

if (SearchFlag)

cout << "Соответствий не найдено" << endl;

}

int main(int argc, char \*argv[]){

setlocale(LC\_ALL, "Russian");

int SelectedValue;

while (true){

cout <<

"Выберите дествие:" << endl <<

" 1. Показать ARP-таблицу" << endl <<

" 2. Добавить запись" << endl <<

" 3. Удалить запись" << endl <<

" 4. Получить MAC по IP" << endl <<

" 5. Завершить" << endl;

cin >> SelectedValue;

switch (SelectedValue){

case 1:

OutputARPTable();

break;

case 2:

AddIpNetEntry();

break;

case 3:

DropIpNetEntry();

break;

case 4:

IPtoMAC();

break;

case 5:

default:

goto Finished;

}

}

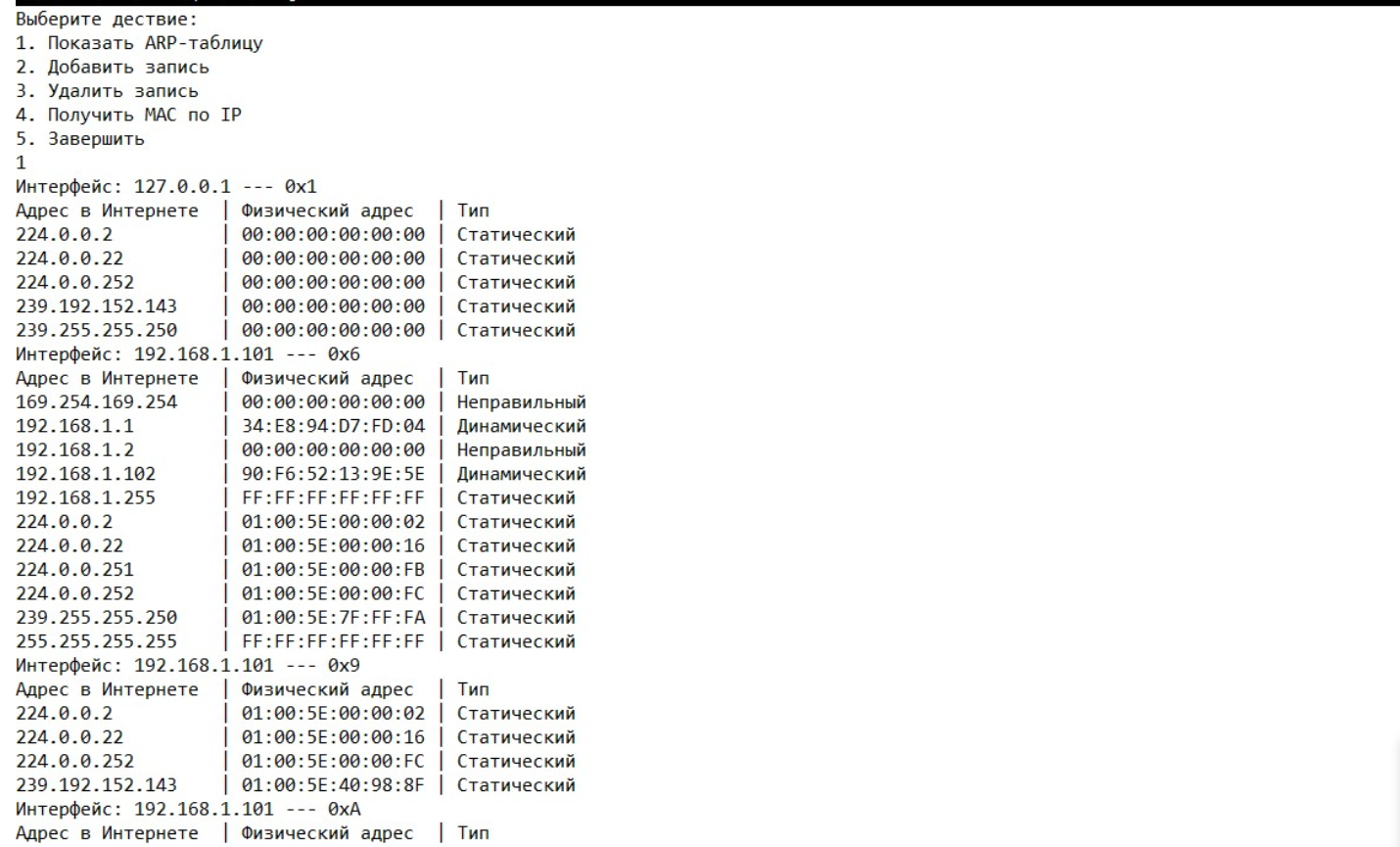
Finished:

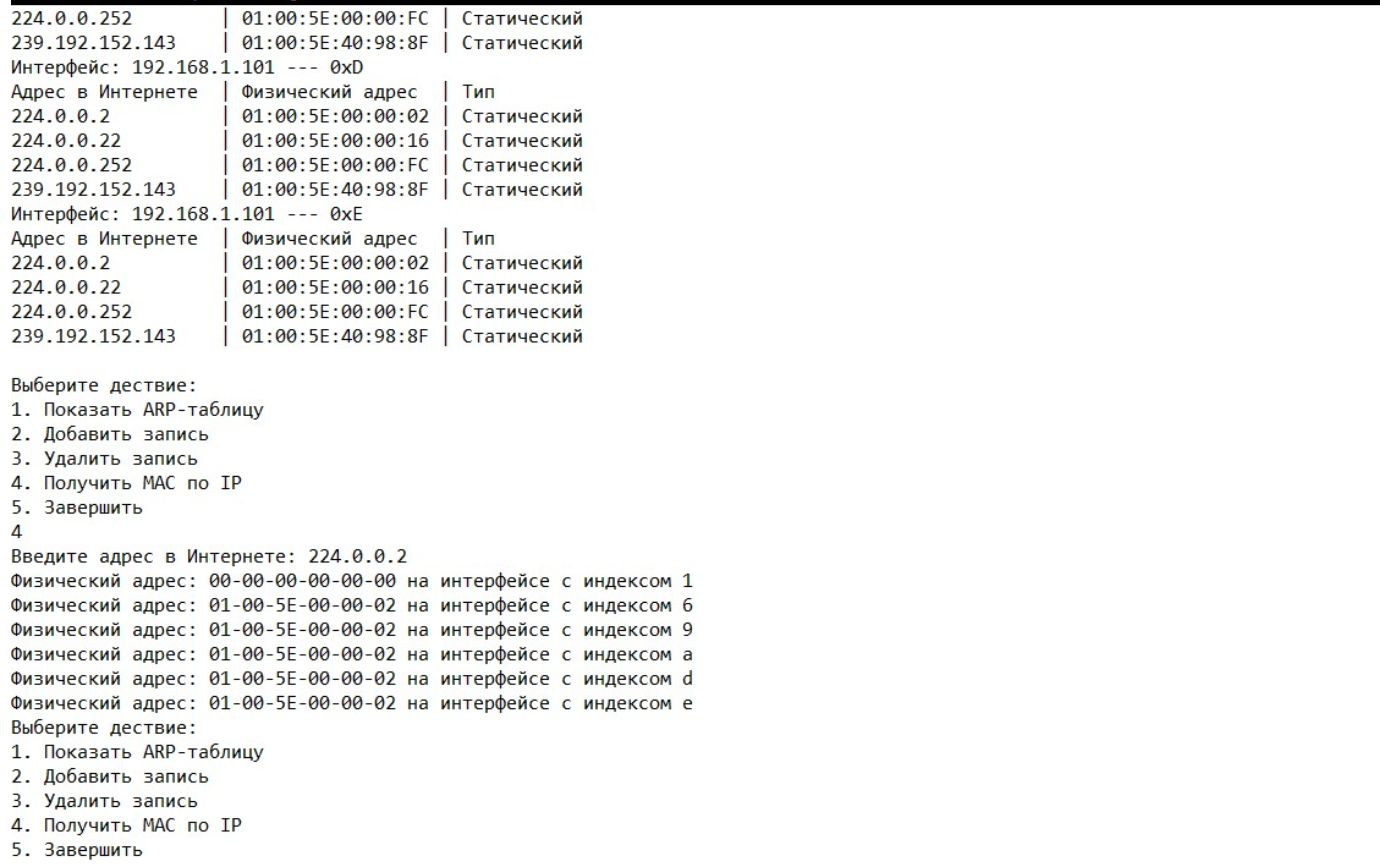
system("PAUSE");

return EXIT\_SUCCESS;

}

**Скриншоты работы программы**





**Вывод**: IP-адреса назначаются независимо от физического аппаратного адреса машины. Чтобы доставить межсетевой пакет, сетевое программное обеспечение должно отобразить IP-адрес в физический аппаратный адрес и использовать этот аппаратный адрес для передачи кадра. Протокол ARP выполняет динамическое разрешение адресов, используя только низкоуровневую сетевую коммуникационную систему. ARP позволяет машинам разрешать адреса, не храня постоянно информации о связках адресов. Каждая машина отвечает на запросы, соответствующие ее IP-адресу, посылая ответы, содержащие требуемый аппаратный адрес. Чтобы сделать ARP эффективным, каждая машина кэширует связки. Так как межсетевой траффик имеет тенденцию состоять из последовательности взаимодействий между парами машин, кэш приводит к ненужности большинства широковещательных запросов ARP.

**Ответы на вопросы**

1. **Какие задачи решает протокол ARP?**

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.

1. **Что такое ARP-таблица? Почему она является необходимым элементом?**

Это буферная память в которой хранятся пары адресов (IP-адрес, физический адрес) устройств в сети. ARP-таблица является обязательной так как наличие она позволяет уменьшить количество ARP-запросов. ARP-таблица необходима потому, что IP-адреса и Ethernet-адреса выбираются независимо, и нет какого-либо алгоритма для преобразования одного в другой.

1. **Типы записей ARP-таблицы.**

В ARP-таблице могут содержаться как статические, так и динамические записи. Динамические записи добавляются и удаляются автоматически, статические заносятся вручную. Кроме того, ARP-таблица всегда содержит запись с физическим широковещательным адресом (OxFFFFFFFFFFFF) для локальной сети. Эта запись позволяет устройству принимать широковещательные ARP-запросы.

1. **Опишите процесс преобразования ip-адреса в локальный.**

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.

1. **Как может работать протокол ARP в глобальных сетях?**

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.

1. **Что представляет собой протокол RARP?**

RARP- это протокол сетевого уровня модели OSI, выполняет обратное отображение адресов, то есть преобразует физический адрес в IP-адрес.

1. **В каких целях может быть использован протокол RARP?**

Протокол RARP - это протокол для нахождения IP-адреса по известному локальному адресу. RARP используется при старте бездисковых станций, не знающих в начальный момент своего IP-адреса, но знающих адрес своего сетевого адаптера