**7. РАЗРЕШЕНИЕ АДРЕСОВ В IP-СЕТЯХ**

**7.2. Определение символьного адреса удаленного хоста по IP-адресу**

***7.2.1. Разработка программы определения имени (DNS, NetBIOS) компьютера по IP-адресу, и наоборот***

**Интерфейс NetBIOS**

Network Basic Input/Output System (NetBIOS) – стандартный интерфейс прикладного программирования (API), разработанный Sytek Corporation для IBM. NetBIOS определяет программный интерфейс для сетевой связи, но не обуславливает физический способ передачи данных по сети. В Win32 интерфейс NetBIOS обеспечивает обратную совместимость со старыми приложениями.

Ключ к пониманию NetBIOS – номера сетевых адаптеров (LAN Adapter, LANA). В первоначальных реализациях NetBIOS каждому физическому сетевому адаптеру присваивалось уникальное значение – номер LANA. В Win32 это стало проблематичным, так как рабочая станция может иметь и множество сетевых протоколов, и множество плат сетевого интерфейса.

Номер LANA соответствует уникальным сочетаниям сетевого адаптера с транспортным протоколом. Так, если рабочая станция имеет две сетевые платы и два поддерживающих NetBIOS транспорта (например, TCP/IP и NetBEUI), будет присвоено четыре номера LANA. Номера LANA лежат в диапазоне от 0 до 9, и операционная система назначает их без какого-либо определенного порядка, кроме LANA 0, который имеет особый смысл – это номер «по умолчанию».

Есть два типа имен NetBIOS – уникальное и групповое. Никакой другой процесс в сети не может зарегистрировать уже имеющееся уникальное имя будет – выдана ошибка дублирования имени. Имена компьютеров в сетях Microsoft – имена NetBIOS. Когда компьютер загружается, он регистрирует свое имя на локальном сервере Windows Internet Naming Server (WINS), который сообщает об ошибке, если другой компьютер уже использует то же имя. Сервер WINS поддерживает список всех зарегистрированных имен NetBIOS.

Вместе с именем могут храниться и сведения о протоколе. Например, в сетях TCP/IP WINS запоминает IP-адрес компьютера, зарегистрировавшего имя NetBIOS.

API-интерфейс NetBIOS содержит только одну функцию:

|  |
| --- |
| UCHAR Netbios(PNCB pNCB); |

Все объявления функций, константы и т.п. для NetBIOS определены в заголовочном файле. Единственная библиотека, необходимая для компоновки приложений NetBIOS – Netapi32.1ib. Добавим в приложение:

|  |
| --- |
| #include <stdio.h>  #include <iostream>  #include <windows.h>  #include <Winsock.h>  #include <Wsnetbs.h>  #pragma comment(lib,"WS2\_32.lib")  #pragma comment (lib, "Netapi32.lib")  using namespace std; |

**Валидация пользовательского ввода**

Перед определением имен и IP-адресов необходимо добавить проверку корректности ввода пользователем IP-нотации адреса и DNS-имени. Для проверки IP добавьте в приложение функцию:

|  |
| --- |
| bool CheckAddr(char\* ip)  {  int points=0, //счетчик точек  numbers=0;//счетчик цифр октета  char\* buff;  buff = new char[3];  for(int i=0;ip[i]!='\0';i++)//просмотреть IP-адрес  {  if(ip[i]<='9'&&ip[i]>='0') //если только цифры  {  if(numbers>3) return false; //не больше 3  buff [numbers++]=ip[i];  //записать в буффер  }  else if(ip[i]=='.') //если следующий октет  {  if(atoi(buff)>255) return false;  //больше 255 – ошибка  if(numbers==0) return false;  //если цифр нет  numbers=0; //обнулить счетчик цифр октета  points++;  buff = new char[3];  }  else return false;  }  if(points!=3) return false;  //если точек меньше 3 - ошибка  if(numbers==0||numbers>3) return false;  return true;  } |

Для проверки верности задания DNS-имени добавьте пользовательскую функцию:

|  |
| --- |
| bool CheckDNSName(char\* dns\_name)  {  for(int i=0;dns\_name[i]!='\0';i++)  if(!(dns\_name[i]>='A'&&dns\_name[i]<='Z'||  dns\_name[i]>='a'&&dns\_name[i]<='z'||  dns\_name[i]>='0'&&dns\_name[i]<='9'||  dns\_name[i]=='.'||  dns\_name[i]=='-')) return false;  //имя может содержать латинские символы, цифры, точки тире  return true;  } |

**Инициализация Winsock**

Инициализируем Winsock следующим образом:

|  |
| --- |
| setlocale(LC\_ALL,"RUS");  WSADATA WsaData;  struct hostent \*dns;  struct hostent \*netbios;  char \*host = new char[16];  int i=0;  bool flag=true;  bool is\_ip=true;  in\_addr addr;  if(WSAStartup(0x0202,&WsaData)==NULL)  cout<<"WSA done!"<<endl; |

Выполняем ввод информации и проверку ввода:

|  |
| --- |
| do  {  if(!flag) cout<<"Неверный IP"<<endl;  cout<<"Введите IP-адрес или DNS-имя: ";  cin>>host; //ввести  if(is\_ip=!isalpha(host[0]))  //если первый - не буква  flag=CheckAddr(host);  //проверить адрес  else  flag=true;  }while(!flag);  if(!is\_ip) //если буква  {  do  {  if(!flag)  {  cout<<"Неверное DNS-имя"<<endl;  cout<<"Введите DNS-имя: ";  cin>>host; //ввести  }  flag=CheckDNSName(host);  //проверить DNS  }while(!flag);  } |

**Поиск сведений об узле**

Если задан IP-адрес, определим DNS и NetBIOS имена. Будем использовать функцию gethostbyaddr, извлекающую информацию о хосте в соответсвии с веденным сетевым адресом. Функция последовательно просматривает файл /etc/hosts с самого начала в поисках имени или адреса главной машины. Формат функции:

|  |
| --- |
| struct hostent\* FAR gethostbyaddr  (  \_\_in const char \*addr,  //указатель на адрес  //в сетевом порядке следования байт  \_\_in int len,  //длина адреса в байтах  \_\_in int type  //тип адреса  ); |

Тип адреса type может принимать значения AF\_INET для семейства адресов IPv4; AF\_NETBIOS для NetBIOS семейства; AF\_INET6 для IPv6.

В случае ошибки возвращается NULL указатель, а специфический код ошибки может быть получен через вызов WSAGetLastError. В случае удачного завершения, функция возвращает структуру hostent, которая используется для хранения информации о заданном хосте:

|  |
| --- |
| typedef struct hostent  { char FAR \*h\_name;  //официальное host-имя  char FAR FAR \*\*h\_aliases;  //список псевдонимов -  //массив альтернативных имен  //машины, оканчивающийся нулем  int h\_addrtype;  //тип адреса  //в настоящее время всегда AF\_INET  int h\_length;  //длина адреса в байтах  char FAR FAR \*\*h\_addr\_list;  //указатель на список адресов хоста,  //оканчивающийся нулем  };HOSTENT, \*PHOSTENT, FAR \*LPHOSTENT; |

Поле h\_name является официальным именем узла. Если в сети используется доменная система имен (DNS), в качестве имени сервера будет возвращено полное имя домена (Fully Qualified Domain Name, FQDN). Если в сети применяется локальный файл узлов (hosts, lmhosts) – это первая запись после IP-адреса. Поле h\_aliases – массив, завершающийся нулем дополнительных имен узла. Поле h\_addrtype представляет возвращаемое семейство адресов. Поле h\_length определяет длину в байтах каждого адреса из поля h\_addrtype. Поле h\_addr\_list – массив, содержащий IP-адреса узла (узел может иметь несколько IP-адресов). Каждый адрес в этом массиве представлен в сетевом порядке. Обычно приложение использует первый адрес из массива. При получении нескольких адресов, приложение должно выбирать адрес случайным образом из числа доступных.

Приложение не может изменять структуру или освобождать поля структуры hostent. Только одна копия данной структуры может размещаться в потоке, поэтому нужную информацию необходимо копировать перед любым другим Windows Sockets API вызовом.

Существует асинхронная версия функции gethostbyaddr  и WSAAsyncGetHostByAddr.

**Разрешение имен**

Если пользователь ввел IP, преобразуем IP-адрес из точечной нотации в 32-битное длинное целое без знака, вызываем функцию gethostbyaddr, получаем указатель на структуру hostent, выводим на консоль официальное host-имя, извлекаем список адрессов хоста для вывода на консоль, используем функцию преобразования inet\_ntoa, конвертирующую IPv4-адрес в ASCII строку десятично-точечной нотации для IP-адресов, и вызываем функцию GetNetBiosName, выводящую имена NetBIOS:

|  |
| --- |
| if(is\_ip)  {  addr.S\_un.S\_addr=inet\_addr(host); //преобразование  dns=gethostbyaddr((char\*)&addr,4,AF\_INET);  if(dns!=NULL) //если структура заполнена,  {  cout<<"DNS имя: "<<dns->h\_name<<endl;  //выводим имя хоста  while(dns->h\_addr\_list[i]!=0)  {  //выодим список адресов хоста  addr.S\_un.S\_addr =  \*(u\_long\*) dns->h\_addr\_list[i++];  cout<<" "<<inet\_ntoa(addr);  }  }  else  {  cout<<"DNS имя не найдено..."<<endl;  }  GetNetBiosName(inet\_ntoa(addr));  } |

На рис. 7.4 приведен результат работы программы.

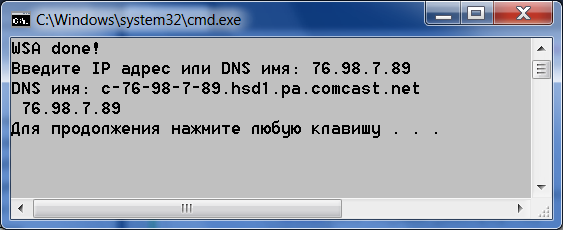


Рис. 7.4. Результат работы программы

В Winsock предусмотрены две функции для разрешения имени в IP-адрес. Функции gethostbyname и WSAAsyncGetHostByName отыскивают в базе данных узла сведения об узле, соответствующие его имени. WSAAsyncGetHostByName – асинхронная версия функции gethostbyname, оповещающая приложение о завершении своего выполнения с помощью сообщений Windows.

Если пользовтель ввел DNS-имя, вызываем функцию gethostbyname, получаем указатель на структуру hostent, выводим на консоль официальное host-имя, извлекаем список адресов хоста для вывода на консоль и вызываем функцию GetNetBiosName:

|  |
| --- |
| else  {  dns = gethostbyname(host);  if(dns!=NULL)  {  cout<<"DNS имя: "<<dns->h\_name<<endl;  while(dns->h\_addr\_list[i]!=0)  {  addr.S\_un.S\_addr =  \*(u\_long\*) dns->h\_addr\_list[i++];  cout<<" "<<inet\_ntoa(addr)<<endl;  }  addr.S\_un.S\_addr =  \*(u\_long\*) dns->h\_addr\_list[0];  GetNetBiosName(inet\_ntoa(addr));  }  else  {  cout<<"DNS имя не найдено..."<<endl;  }  } |

На риc. 7.5 приведен результат работы программы.

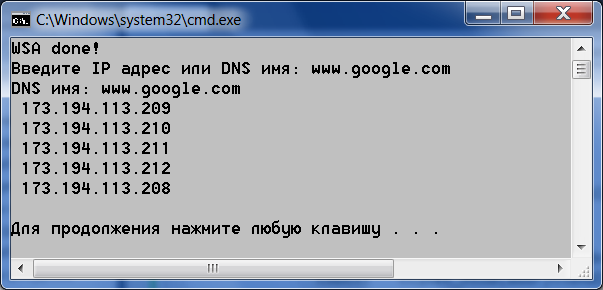


Рис. 7.5. Результат работы программы

**API-интерфейс NetBIOS и блок сетевого управления (NCB)**

Функция GetNetBiosName определяет NetBIOS имена компьютера по IP через стандартную функцию Netbios. Наиболее важная особенность функции Netbios – параметр pNCB, который является указателем на блок сетевого управления (network control block, NCB). Это указатель на структуру NCB, содержащую всю информацию, требуемую функции Netbios для выполнения команды. Определяется эта структура так:

|  |
| --- |
| typedef struct \_NCB  {  UCHAR ncb\_command;  UCHAR ncb\_retcode;  UCHAR ncb\_lsn;  UCHAR ncb\_num;  PUCHAR ncb\_buffer;  WORD ncb\_length;  UCHAR ncb\_callname[NCBNAMSZ];  UCHAR ncb\_name[NCBNAMSZ];  UCHAR ncb\_rto;  UCHAR ncb\_sto;  void (CALLBACK \*ncb\_post)( struct \*NCB);  UCHAR ncb\_lana\_num;  UCHAR ncb\_cmd\_cplt;  UCHAR ncb\_reserve[X];  HANDLE ncb\_event;  } NCB, \*PNCB; |

He все члены структуры будут использоваться в каждом вызове Netbios; некоторые из полей данных являются выходными параметрами (другими словами, задаются по возвращении из вызова Netbios). Важно всегда обнулять структуру NCB до вызова Netbios. Это можно сделать функцией:

ZeroMemory(&ncb,sizeof(NCB));

Рассмотрим состав структуры NCB:

ncb\_command – указывает выполняемую команду Netbios. Многие команды могут выполняться синхронно или асинхронно;

ncb\_retcode – определяет код возврата для данной операции;

ncb\_lsn – определяет номер локального сеанса, уникально идентифицирующий его в текущем окружении. Функция возвращает новый номер сеанса после успешной команды NCBCALL (открыть сессию с другим именем) или NCBLISTEN;

ncb\_num – указывает номер локального сетевого имени. Новый номер возвращается для каждого вызова команды NCBADDNAME (добавить новое имя) или NCBADDGRNAME (добавить новое групповое имя);

ncb\_buffer – указывает на буфер данных. Для команд, которые отправляют данные, этот буфер содержит отправляемые данные; для команд, которые получают данные – данные, возвращаемые функцией Netbios. Для других команд, типа NCBENUM (перечисление адаптеров LAN), буфер будет предопределенн структурой LANA\_ENUM;

ncb\_length – указывает длину буфера в байтах. Для команд приема присваивает этому полю значение, равное количеству полученных байтов, если буфер недостаточно велик, Netbios возвращает ошибку;

ncb\_callname – указывает имя удаленного приложения;

ncb\_name – указывает имя, под которым известно приложение;

ncb\_rto – указывает время ожидания (тайм-аут) для операций приема (значение определено в 500-миллисекундных единицах);

ncb\_sto – указывает время ожидания для операций отправки;

ncb\_post – указывает адрес процедуры, которую надо вызвать по завершении асинхронной команды. Функция определена как void CALLBACK PostRoutine(PNCB pncb);

ncb\_lana\_num – указывает номер LANA для выполнения команды;

ncb\_cmd\_cplt – определяет код возврата для операции;

ncb\_reserve – зарезервировано, должно быть равно 0;

ncb\_event – указывает описатель объекта события Windows в свободном (nonsignaled) состоянии. Когда асинхронная команда завершается, событие переходит в занятое (signaled) состояние. Следует использовать только ручной сброс событий.

**Пример определения имен NetBIOS**

Теперь рассмотрим текст функции GetNetBiosName. Она получает все номера LANA, для каждого LAN извлекает статус адаптера, если определено имя NetBios, выводит на консоль:

|  |
| --- |
| int GetNetBiosName(char \*dwIpAddr)  {  ADAPTER\_STATUS \*pStatus; //Статус адаптера  NAME\_BUFFER \*pNames; //Структура имя NetBios  NCB ncb; //Структура NetBios  HANDLE hHeap;  WORD cbBuffer;  char \*Addr;  UCHAR rc;  int i;  //Получаем IP-адрес в строковом виде  in\_addr in;  if (dwIpAddr==0) return 0;  in.s\_addr = inet\_addr(dwIpAddr);  Addr=inet\_ntoa(in);  if (!Addr) return 0;  //Получить все номера LANA  LANA\_ENUM lan\_num; //перечисление всех lana  //Инициализация/очистка структуры NCB  ZeroMemory(&ncb,sizeof(NCB));  ncb.ncb\_command = NCBENUM;  //определить код для заполнения LANA\_ENUM структуры  ncb.ncb\_buffer = (unsigned char \*) &lan\_num;  ncb.ncb\_length = sizeof(lan\_num); //размер  rc=Netbios (&ncb); //вызвать функцию  for (i=0;i < lan\_num.length; i++)  //для каждого LANA  {  //Инициализация/очистка  ZeroMemory(&ncb,sizeof(NCB));  //Сброс всех сведений о LANA, перечисленных в структуре  //LANA\_ENUM и использование первого NetBIOS-имени  ncb.ncb\_command = NCBRESET;  ncb.ncb\_lana\_num = lan\_num.lana[i];  //установить номер LANА  rc=Netbios (&ncb); //вызвать функцию  hHeap = GetProcessHeap();  //буфер под статус адаптера + список имен(255 имен)  cbBuffer = sizeof (ADAPTER\_STATUS)  + 255 \* sizeof (NAME\_BUFFER);  //выделить блок памяти для кучи и инициализирвать 0  pStatus = (ADAPTER\_STATUS \*) HeapAlloc (hHeap,HEAP\_ZERO\_MEMORY,cbBuffer);  if (pStatus==NULL) return 0;  //Инициализация/очистка структуры NCB  ZeroMemory(&ncb,sizeof(NCB));  ncb.ncb\_command = NCBASTAT;  //извлечь статус адаптера  ncb.ncb\_lana\_num = lan\_num.lana[i];//для i LANA  ncb.ncb\_buffer = (PUCHAR) pStatus;  ncb.ncb\_length = cbBuffer;  strcpy((char \*)ncb.ncb\_callname,Addr);//для IP  rc = Netbios(&ncb); //вызвать функцию  if (ncb.ncb\_retcode==NRC\_GOODRET)  //если определено имя NetBios  {  pNames = (NAME\_BUFFER \*)(pStatus + 1);  printf("\n Имена NetBios:\n");  for ( i = 0;i < pStatus->name\_count;i++)  {  //вывести имя на консоль  pNames->name[15]='\0';  printf("%d: %s\n", i+1, pNames->name);  pNames++;  }  }  HeapFree(hHeap,0,pStatus);  }  return 0;  } |

В конце программы не забудьте вызвать функцию WSACleanup().

***7.2.2. Лабораторная работа №11-12***

**Цель:** Изучение методов определения символьного (DNS, NetBIOS) адреса удаленного хоста, и наоборот.

**Задание:** Лабораторная работа заключается в разработке программы, выполняющей определение символьного (DNS, NetBIOS) адреса по введенному IP-адресу, и наоборот, для удаленного хоста по сети.