**北京邮电大学课程设计报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **课程设计**  **名称** | 计算机网络课程设计 | | **学 院** | 计算机 | **指导教师** | **蒋砚军** |
| **班 级** | **班内序号** | **学 号** | | **学生姓名** | **成绩** | |
| **2020211302** | **15** | **2020211262** | | **鄭毓恒** |  | |
| **2020211302** | **14** | **2020211261** | | **赵维岗** |  | |
| **2020211302** | **23** | **2020211275** | | **丁易** |  | |
| **课**  **程**  **设**  **计**  **内**  **容** | 简要介绍课程设计的主要内容，包括课程设计教学目的、基本内容、实验方法和团队分工等  课程设计教学目的：设计一个DNS服务器程序，利用对DNS的知识进行实践。  基本内容：设计一个DNS服务器程序，读入“域名-IP地址”对照表，当客户端查询域名对应的IP地址时，用域名检索该对照表，得到三种检索结果，实现不良网站拦截功能、服务器功能和中继功能。  实验方法：C语言编程  团队分工：  鄭毓恒：实验代码的设计和编写  赵维岗：实验代码的调试和实验报告的编写  丁易：实验代码的设计和实验报告的编写 | | | | | |
| **学生**  **课程设计**  **报告**  （附页） |  | | | | | |
| **课**  **程**  **设**  **计**  **成**  **绩**  **评**  **定** | **评语**:  **成绩**:  指导教师签名：  年 月 日 | | | | | |

注：评语要体现每个学生的工作情况，可以加页。



**计算机网络课程设计实验报告**



**题目：DNS中继服务器程序的设计与实现**

**班 级： 2020211302班\_\_\_**

**成 员： 2020211262 鄭毓恒**

**2020211275 丁 易**

**2020211261 赵维岗**

**学 院： 计算机学院\_\_\_\_\_**

**2022年 6 月 7 日**

**一、系统功能设计**

**功能说明**

DNS（Domain Name System，域名系统），因特网上作为域名和IP地址互相映射的一个分布式数据库，能够使用户更方便的访问互联网，而不用去记住能够被机器直接读取的IP数串。通过主机名，最终得到该主机对应的IP地址的过程叫做域名解析（或主机名解析）。DNS协议运行在UDP协议之上，使用端口号53。

对于每一级域名长度的限制是63个字符，域名总长度则不能超过253个字符。

本次课程设计要求实现一个DNS中继服务器，读入“域名-IP地址”对照表，当客户端查询域名对应的IP地址时，用域名检索该对照表，三种检索结果：

1. 检索结果为ip地址0.0.0.0，则向客户端返回“域名不存在”的报错消息，而不是返回IP地址为0.0.0.0（不良网站拦截功能）

2. 检索结果为普通IP地址，则向客户返回这个地址（服务器功能）

3. 表中未检到该域名，则向因特网DNS服务器发出查询，并将结果返给客户端（中继功能）

**功能分析**

（1）DNS报文格式

+---------------------+

| Header |

+---------------------+

| Question | the question for the name server

+---------------------+

| Answer | RRs answering the question

+---------------------+

| Authority | RRs pointing toward an authority

+---------------------+

| Additional | RRs holding additional information

+---------------------+

（2）Header报头字段格式

1 1 1 1 1 1

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| ID |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

|QR| Opcode |AA|TC|RD|RA| Z | RCODE |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| QDCOUNT |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| ANCOUNT |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| NSCOUNT |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| ARCOUNT |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

ID: 由客户程序设置并由服务器返回结果。客户程序通过它来确定响应与查询是否匹配

QR: 0表示查询报，1表示响应报。

OPCODE: 通常值为0（标准查询），其他值为1（反向查询）和2（服务器状态请求）。

AA: 权威答案(Authoritative answer)

TC: 截断的(Truncated)应答的总长度超512字节时，只返回前512个字节

RD: 期望递归(Recursion desired)查询报中设置，响应报中返回。告诉名字服务器处理递归查询。如果该位为0，且被请求的名字服务器没有一个权威回答，就返回一个能解答该查询的其他名字服务器列表，这称为迭代查询

RA: 递归可用(Recursion Available)如果名字服务器支持递归查询，则在响应中该比特置为1。

Z: 必须为0，保留字段

RCODE: 响应码(Response coded)，仅用于响应报。值为0(没有差错)，值为3表示名字差错。从权威名字服务器返回，表示在查询中指定域名不存在

QDCOUNT: 问题字段数

ANCOUNT: 回答记录字段数

NSCOUNT: 权威记录字段数

ARCOUNT: 附加记录字段数

（3）Question字段格式

1 1 1 1 1 1

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| |

/ QNAME /

/ /

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| QTYPE |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| QCLASS |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

QNAME: 域名，例如www.bupt.edu.cn

QTYPE: 请求类型，例如A(1)、MX(15)、CNAME(5)、PTR(12)

QCLASS: 请求类别，例如IN(1)

（4）资源记录（RR）字段格式

1 1 1 1 1 1

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| |

/ /

/ NAME /

| |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| TYPE |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| CLASS |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| TTL |

| |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

| RDLENGTH |

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--|

/ RDATA /

/ /

+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

NAME: 名字

TYPE: RR的类型码，例如A(1)、MX(15)、CNAME(5)、PTR(12)

CLASS: 通常为IN(1)，指Internet数据

TTL: 客户程序保留该资源记录的秒数，稳定的资源记录生存时间值可以为2天，它确定了客户端DNS cache可以缓存该记录多长时间

RDLENGTH: 资源数据长度。说明资源数据的字节数，对类型1（TYPE A记录）资源数据是4字节的IP地址

RDATA: 资源数据

**主要功能说明**

（1）DNS服务功能

将计算机的DNS服务器设置为本地之后，当用户需要通过域名查询IP时，DNS服务器将从存储在本地的域名-IP映射表和Cache中进行查询，将查询到的IP地址返回给客户端，从而根据IP地址进行网页访问

（2）DNS中继服务功能

当本地DNS服务器的域名-IP对照表中没有存储所需转换的域名时，则向外部的DNS服务器发出查询请求，并将查询结果返回到客户端，并将其加入到本地服务器的Cache中以便于下次查询时使用

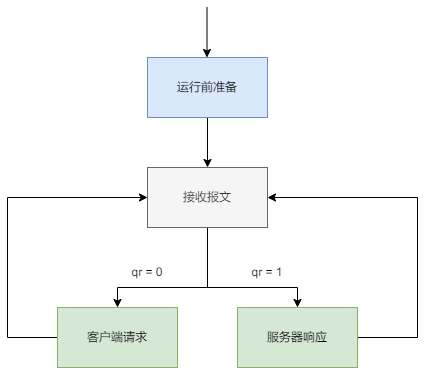
（3）DNS屏蔽功能

当检索对照表得到IP地址的结果为0.0.0.0时，将返回信息的RCODE设置为3，以表示域名不存在

**二、模块划分与软件流程**

**1. 整体流程**

程序的总体模块划分如下图所示：



可以看到运行程序运行的过程中总共有三个模块，分别为运行前的准备、客户端请求和服务器响应，其中运行前的准备主要是为了程序的运行将一些程序中需要用到的数据结构进行初始化以及对文件的写入，客户端请求用于处理客户端发送的查询报文，服务器响应用于处理在向服务器发送查询之后服务器端返回的结果，并将其返回原来的 客户端。下面将对这三个功能模块做进一步说明，在此之前需要先对程序中使用的 数据结构进行解释。

**2.数据结构**

（1）DNS报文首部

typedef struct

{

unsigned id : 16; /\* query identification number \*/

unsigned rd : 1; /\* recursion desired \*/

unsigned tc : 1; /\* truncated message \*/

unsigned aa : 1; /\* authoritive answer \*/

unsigned opcode : 4; /\* purpose of message \*/

unsigned qr : 1; /\* response flag \*/

unsigned rcode : 4; /\* response code \*/

unsigned cd : 1; /\* checking disabled by resolver \*/

unsigned ad : 1; /\* authentic data from named \*/

unsigned z : 1; /\* unused bits, must be ZERO \*/

unsigned ra : 1; /\* recursion available \*/

uint16\_t qdcount; /\* number of question entries \*/

uint16\_t ancount; /\* number of answer entries \*/

uint16\_t nscount; /\* number of authority entries \*/

uint16\_t arcount; /\* number of resource entries \*/

}DNSheader;

报文首部的设计参考实验说明资料中的做法，使用一个结构体来保存报文首部的相关信息，其中id字段由客户程序设置并由服务器返回结果，客户程序通过它来确定响应与查询是否匹配，qr字段用于确定查询报（0）与响应报（1），opcode字段通常值为0（标准查询），其他值为1（反向查询）和2（服务器状态请求），aa字段为权威答案，tc字段为当应答的总长度超512字节时，只返回前512个字节，rd为期望递归，查询报中设置，响应报中返回，同时告诉名字服务器处理递归查询。如果该位为0，且被请求的名字服务器没有一个权威回答，就返回一个能解答该查询的其他名字服务器列表，这称为迭代查询，ra字段为递归可用，如果名字服务器支持递归查询，则在响应中该比特置为1,z为保留字段，必须设置为0，rcode为响应码，仅用于响应报，当其值为0时表示没有差错，值为3表示名字差错，从权威服务器返回，表示在查询中指定域名不存在。

（2）DNS Question字段

typedef struct

{

unsigned char qname[MAX\_NAME\_LEN];

uint16\_t qtype;

uint16\_t qclass;

}DNSquestion;

其中qname为一个字符数组，用于存储域名，qtype为用octet码表示的查询类型，qclass也是类似的。

（3）DNS RR字段

typedef struct

{

uint16\_t name;

uint16\_t type;

uint16\_t \_class;

uint16\_t ttl;

uint16\_t \_ttl;

uint16\_t rdlen;

uint32\_t rdata;

}DNSrr;

其中name为名字，type为RR的类型码，\_class通常为IN(1)，指Internet数据，ttl为客户程序保留该资源记录的秒数，稳定的资源记录生存时间值可以为2天，它确定了客户端DNS cache可以缓存该记录多长时间，rdlen为资源数据长度，说明资源数据的字节数，对类型1（typeA记录）资源数据是4字节的IP地址，rdata为资源数据

（4）解析项

解析项中有IP地址以及对应域名，用于存储本地数据

typedef struct

{

uint32\_t ip;

char\* domain;

}Resolve;

单链表节点形式的解析项，用于存储本地数据

typedef struct snode

{

Resolve item;

struct snode\* next;

}resolveSingleP;

双向链表节点的解析项，用于存储cache

typedef struct dnode

{

unsigned char\* domain;

unsigned char response[BUF\_SIZE];

struct dnode\* prev;

struct dnode\* next;

}resolveTwowayP;

（5）本地文件数据存储

typedef struct

{

resolveSingleP\* localHead;

int itemNum;

}LocalRecord;

使用单链表结构存储本地文件数据，itemNum存储数据的数目，使用之前定义的单链表解析项构造单链表存储

（6）Cache

typedef struct

{

resolveTwowayP\* cacheHead;

int itemNum;

}Cache;

与本地数据类似地，itemNum存储数据的数目，使用之前定义的双链表解析项构造双向链表存储

（7）Query

typedef struct

{

char\* domain;

uint16\_t oldID;

SOCKADDR\_IN client;

}Query;

Query用于存储等待响应的查询，其中domain用于存储域名，oldID用于存储原始的ID，client用于存储原始查询客户端的socket地址

**3.模块划分**

（1）运行前准备

在程序运行之前需要对程序中使用的数据结构以及一些变量进行定义与初始化，这些准备工作可以作为一个模块进行说明。

在程序运行之初首先需要进行参数的处理，设置了三种输入参数，以实现不同的功能设置，用法如下所示

dnsrelay [-s dns-server-ipaddr] [-f filename]

-h (输入帮助)

-s (使用指定DNS服务器，否则默认使用DNS服务器)

-f (使用指定路径的DNS配置文件，否则默认使用当前目录下dnsrelay.txt)

使用parsing\_parameters(argc, argv)函数对输入的参数进行处理

然后运行initSock()和initCache()函数，前者对socket进行设置，包括初始化socket库、检测版本号、创建socket、将套接口设置为非阻塞，并进行DNS服务器地址信息和本地地址信息的初始化以及对端口进行绑定，后者对Cache进行初始化。

接着读取本地的域名-IP映射表，最后创建两个socket地址，其中from为发送方的socket地址，clientTo为在接收到服务端的报文之后将该报文进行转发的目的socket地址。

（2）客户端请求

在客户端请求中可以分为标准查询和非标准查询两种查询格式，对于两种查询格式分别进行处理。

对于标准查询，首先调用getQuestionSection(&question, &nameLen)函数获取Question字段，然后调用octet2DomainName(question.qname, nameLen, domain)将octet码形式的域名转换为ASCII格式，完成之后在首先在本地数据中查询域名，如果查询成功，首先将报文头部的qr字段设置为1，更改为响应报文，然后判断查询的结果，假如IP结果为0.0.0.0，则该域名为需要屏蔽的不良网站，将头部的rcode改为3，并设置发送缓冲区的内容，如果不是需要屏蔽的网站，则制作RR字段并更改首部的相关信息，并设置发送缓冲区的内容。如果在本地数据查询失败则在Cache中查找，在Cache中命中之后，将报文更改为响应报文，除去ID部分之外其余部分只需复制Cache中的response即可，并将其发送给客户端。如果在Cache和本地数据中均找不到结果，则将其暂存到query中等待服务器的响应。对于非标准查询，同样将其暂存在query中，并在标准判断结束之后调用sendDNS(serverAddr)函数将其发送给外部的服务器进行处理。

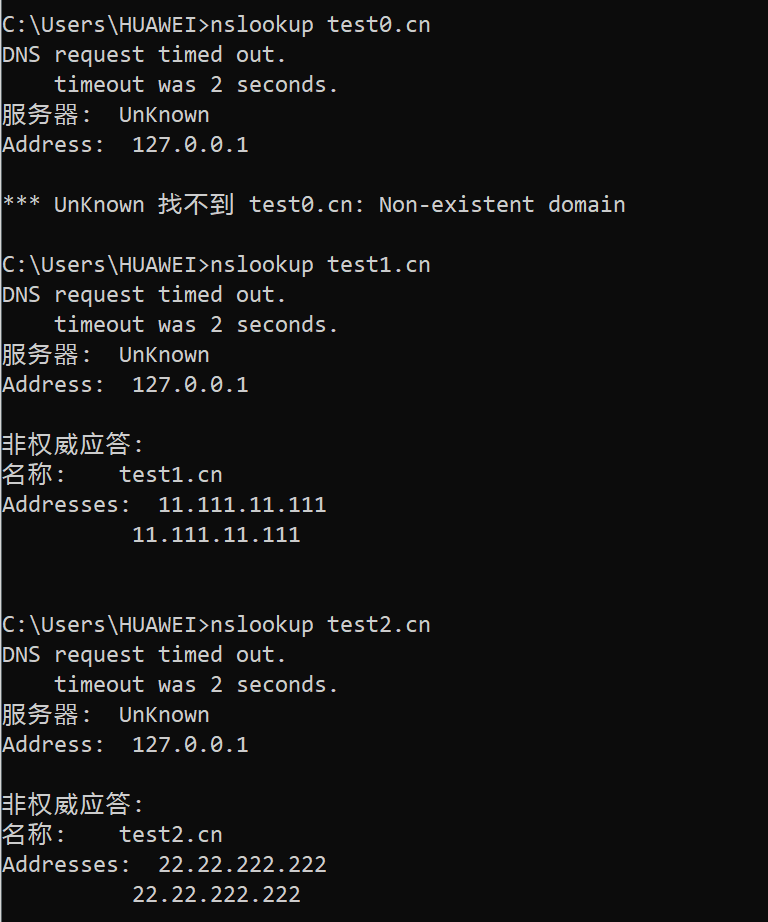
（3）服务端请求

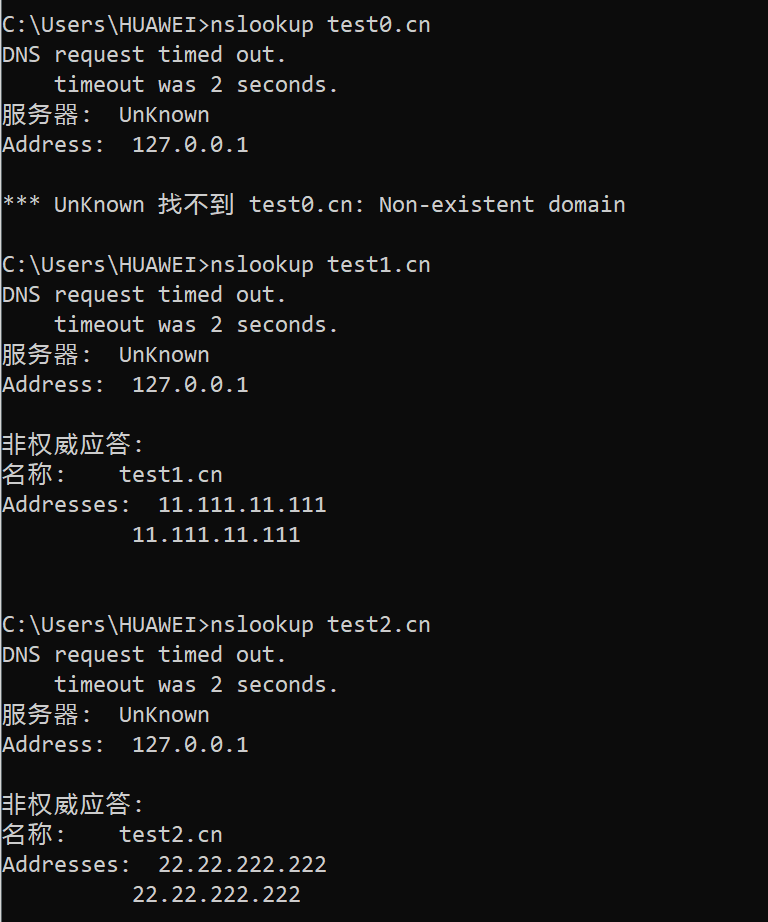
对于服务器的响应，首先需要获取原来的ID和客户端的地址，并将首部的ID进行更改，然后判断该响应是否为标准查询，如果是新的响应则将其添加到Cache中，并输出相关信息，如果是非标准查询的响应则直接输出相关信息，最后将报文转发至客户端。

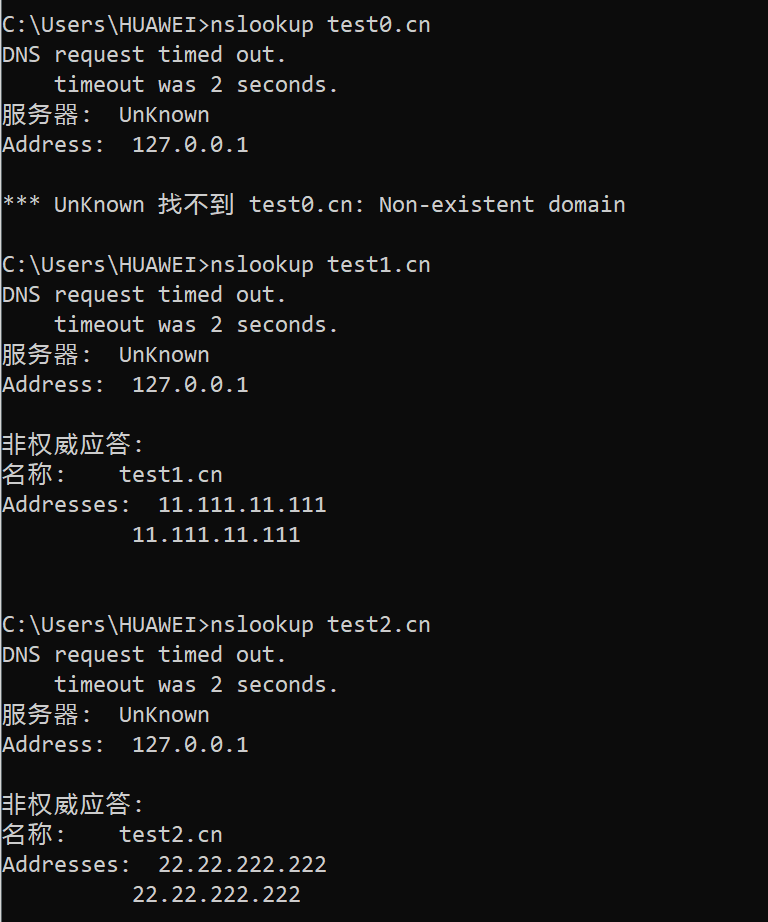
**三、测试用例与运行结果**

将本机的DNS服务器设置为127.0.0.1，在terminal中输入.\dnsrelay.exe之后进行测试

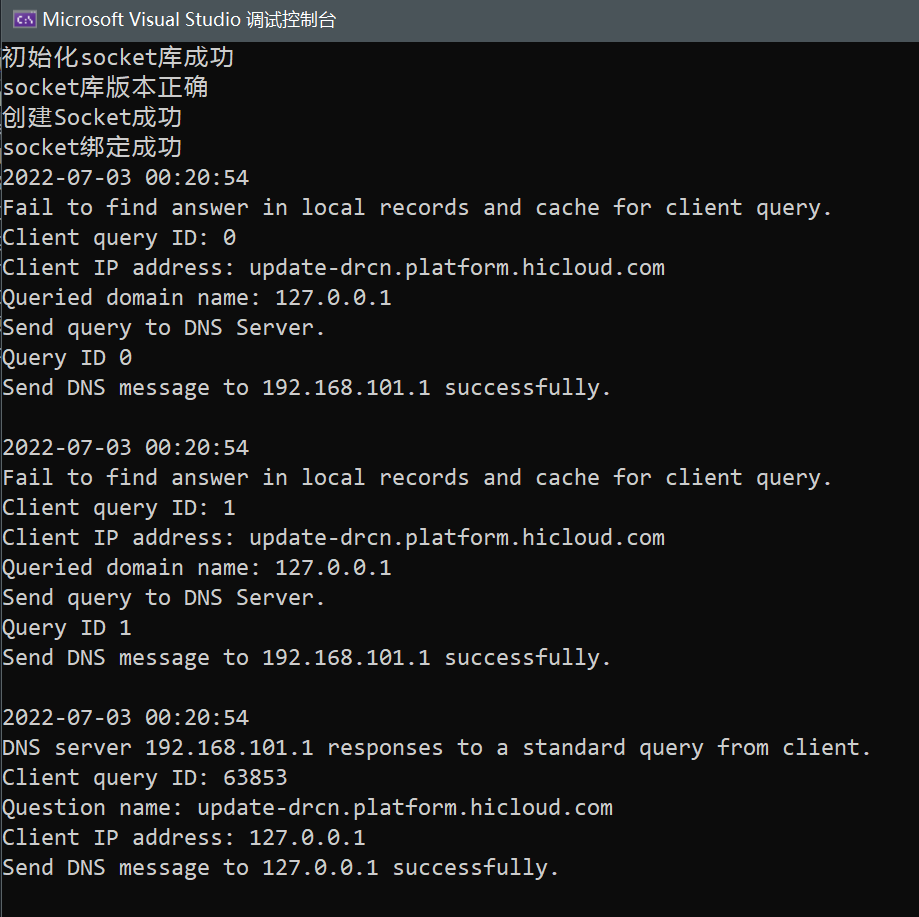
测试nslookup命令，分别使用test0.cn, test1.cn, test2.cn，输出的结果如下所示，符合预期的效果







同时主机可以进行正常的上网，在程序中将输出相关信息，测试时的结果如下图所示



**四、调试遇到的问题与解决**

在调试过程中，遇到了许多问题。

代码中使用了inet\_ntop()、inet\_pton()等函数，一开始无法通过编译，表示这些函数没有定义。查找资料后发现，应该include Ws2tcpip.h头文件，同时要对文件的头文件包含和连接库的顺序进行更改，改为将以下三行放到代码的最开始。

#include <WinSock2.h>

#include <Ws2tcpip.h>

#pragma comment(lib, "WS2\_32.lib")

在编写获取命令行参数的部分代码时，需要使用getopt.h头文件中的getopt()函数，而我们成员的电脑都没有该头文件，因此从网上下载了一份getopt.h头文件，与课设代码放在同一文件夹下。

在对DNS报文的Question部分的Qname字段的octet码进行解码时，发现解码的结果域名是xxx.xxx.xxx.格式，而读取dnsrelay.txt文件时将域名以xxx.xxx.xxx格式存储，区别在于最后有无’.’。最后选择以无’.’作为统一格式，将每次解码的结果的最后一个’.’改为’\0’。

在存储本地文件信息的静态表中找到答案后，制作响应DNS报文时使用了自定义的存储DNS Resource Record部分的数据结构，需要用四个字节存储TTL字段，一开始我们认为可以使用uint32\_t类型进行存储。后续运行程序发现，客户端会得到一个统一的错误答案。查找资料后发现，TTL字段应用两个uint16\_t进行存储。原因是TTL字段之前使用了三个uint16\_t类型，若TTL字段使用uint32\_t类型，结构体将会为了对齐而进行调整，导致报文出错，因此需要使用两个uint16\_t。

**五、总结与心得体会**

通过本次实验，通过设计与实现这一DNS中继服务器，对于DNS的功能以及实现有了比较细致的理解，将理论的概念落到实地。通过在程序中对socket的使用，增强了对于socket编程的理解，同时对于网络编程的经验有了进一步的积累。在程序中需要大量使用C语言中的结构体和指针，在这些应用中对于基础知识的把握更加深刻，同时使用了单链表和双向链表，学习到了相较于传统方式更加高效的链表构造方式，对于数据结构的知识有了新的认识。