2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC)

Design and Implementation of Traditional DNS Protocol

Qi Li, Xingqun Qi, Jiaming Liu, Hongfeng Han*
International school, Beijing University of Posts and Telecommunications
Beijing 100876, China;
* Corresponding Author: hanhongfeng@bupt.edu.cn

***Abstract*—Hosts need to identify their location on the Internet. One way is to use an IP address. In IPV4 networks, each computer is identified by a unique 4-byte number, which is generally written as a four-point format, such as 199.1.35.90. When data are transmitted via the network, the header of the packet will contain the IP address of the destination host. Another method is to use the host name to identify themselves, such as www.yahoo.com, www.google.com. Compared with the IP address, it is more memorable. However, the host name provides little or no information about the host's location in the network. In addition, since host names are usually composed of alphanumeric characters of varying lengths, routers are better at handling fixed-length, hierarchical IP addresses. Therefore, domain name system (DNS) has been developed for converting host names that are easy to be remembered by humans into digital Internet addresses. But the use of DNS sometimes reduces the efficiency of accessing the network because the network environment varies all the time, which results in the transmission rate of information between the client and the DNS server. If the package losses due to timeout, the respond of the DNS server cannot be received. As a result, this paper designs a model that can decrease the probability of these problems by introducing a local DNS database.**

Keywords-Host; IP; DNS; local database (key words)

I. Introduction

Until now, many sites still use DNS load balancing to keep their websites addressable and operating well. DNS's technology is more flexible and convenient. Besides, it is cheap and suitable for most TCP / IP applications[1]. For Web applications, it is unnecessary to make any changes for the code. In response to the huge amount of traffic, DNS uses caching and distributed databases. However, DNS protocol has some detects. The DNS server evenly distributes Http requests to the backend Web server regardless of the current load on each Web server. If the configuration and processing power of different backend Web server are unequal, the slowest Web server will become the bottleneck of the system, servers with strong processing power can not fully play their role. Therefore, building a local DNS database can effectively improve the efficiency of the DNS protocol. This can also reduce the situation client requests concentrated on a single server.

II. Background

A. Port.

Modern computers run different processes at the same time. For example, an email process needs to be separated from an FTP request, and FTP is separated from Web flow. These are all achieved through the port. Each computer has nearly a few thousand logical ports, each transport layer protocol has 65535 ports. These are just abstractions in computer memory, unlike physical entities like USB ports. Each port is identified by a decimal number between 1 and 65535, and it can be assigned as a specific task. For instance, the underlying HTTP protocol for the Web generally uses port number 80[2].

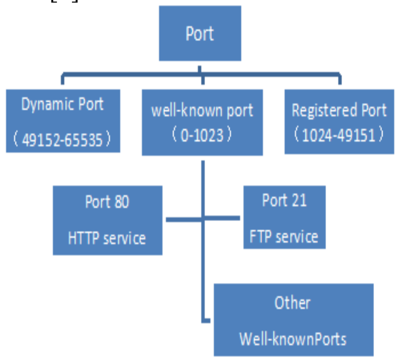


Figure 1. Classifications of port

B. User Datagram Protocol.

TCP / IP network provides two different transport protocols for the application. One is User Datagram Protocol (UDP), which is a lightweight transport protocol that does not provide unnecessary service, only process-to-process data delivery and error checking are available. UDP provides unreliable, connectionless service for the application process that invokes it, so there is no handshake before two processes communicating. Unlike TCP, when a process sends a message into a UDP socket, the UDP protocol does not ensure the message to reach the destination process of the receiver completely. Besides, the packets that arrive at the receiving process may be out of order[3].

1) UDP port number of DNS

DNS is a distributed database implemented by a tiered DNS server, also an application layer protocol that enables hosts to query distributed databases. DNS resolution is the actual addressing method for the vast majority of Internet applications. It operates on UDP, using port 53, which is mainly used for domain name resolution. Applications run on the end system (host) need to convert the host name to an IP address. So these hosts will call the DNS client and send the host name that needs to be converted. After the host name is received by DNS on the user host, a DNS query message will be generated and sent to the network. All DNS request and reply messages are sent via this port using UDP datagrams. The re-development of domain name technology and the wide variety of applications based on domain name technology have enriched Internet applications and protocols[4].

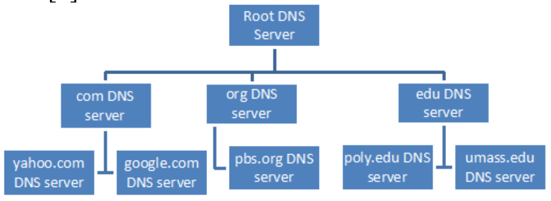
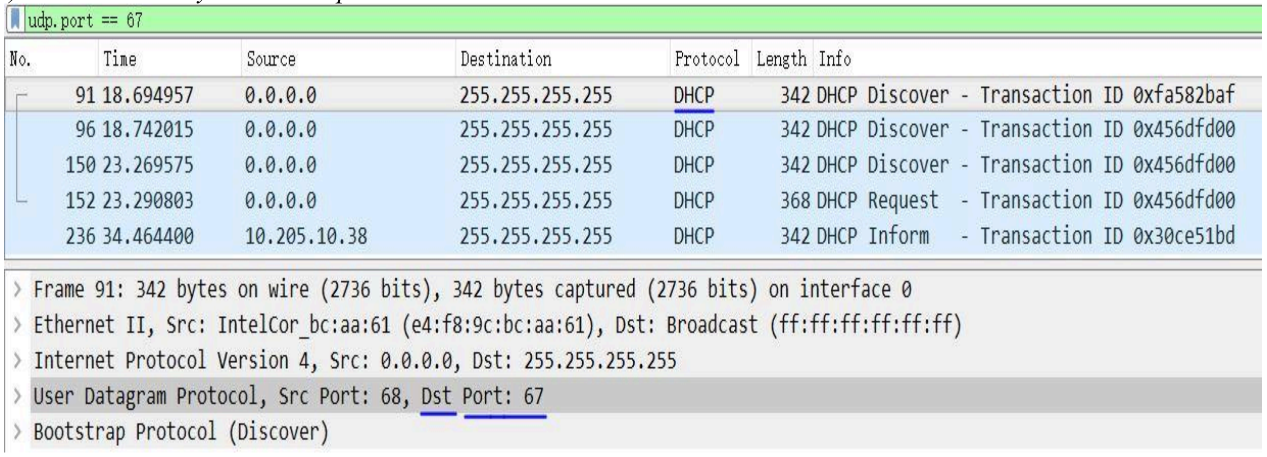


Figure 2. DNS Server

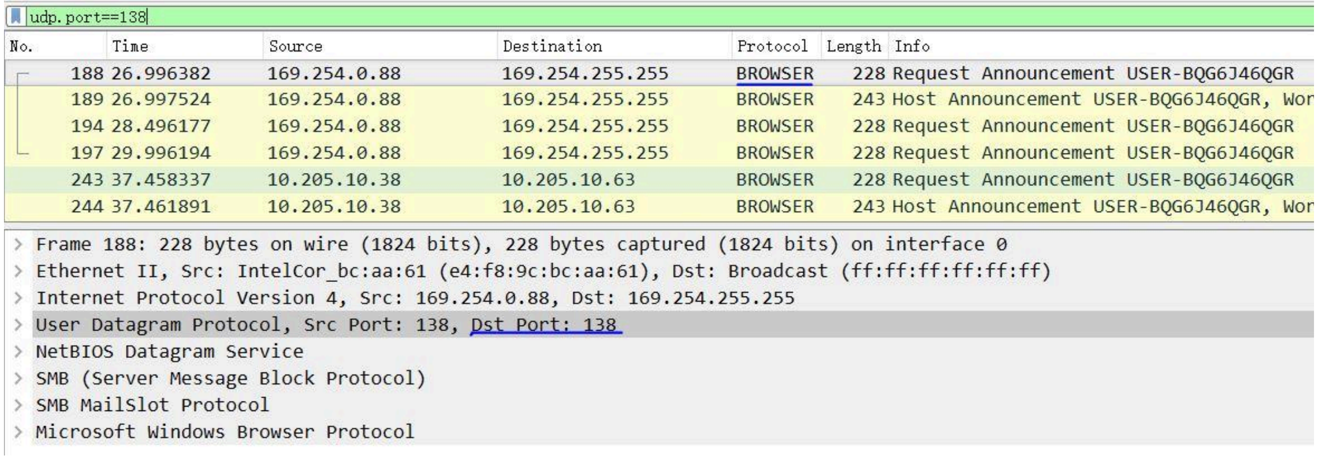
2) Other commonly used UDP port



```
udp.port == 67
No.   Time        Source        Destination       Protocol  Length  Info
91    18.694957   0.0.0.0       255.255.255.255   DHCP      342     DHCP Discover - Transaction ID 0xfa582baf
96    18.742015   0.0.0.0       255.255.255.255   DHCP      342     DHCP Discover - Transaction ID 0x456dfd00
150   23.269575   0.0.0.0       255.255.255.255   DHCP      342     DHCP Discover - Transaction ID 0x456dfd00
152   23.290803   0.0.0.0       255.255.255.255   DHCP      368     DHCP Request  - Transaction ID 0x456dfd00
236   34.464400   10.205.10.38  255.255.255.255   DHCP      342     DHCP Inform   - Transaction ID 0x30ce51bd

> Frame 91: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
> Ethernet II, Src: IntelCor_bc:aa:61 (e4:f8:9c:bc:aa:61), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
> User Datagram Protocol, Src Port: 68, Dst Port: 67
> Bootstrap Protocol (Discover)
```

Figure 3. UDP port 67

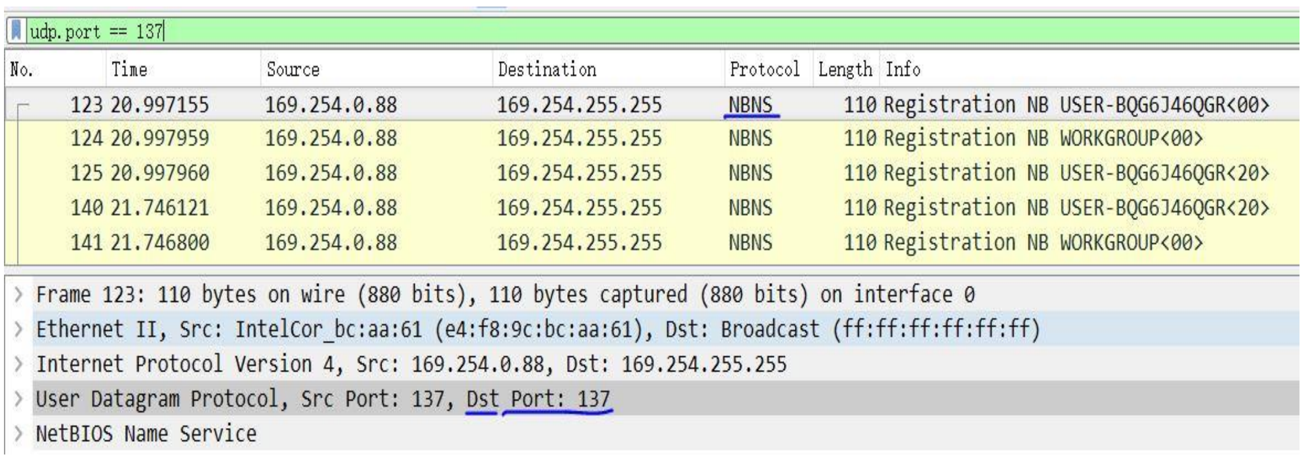
Port 67 Description: Dynamic Host Configuration Protocol (DHCP), which is a local area network protocol that is primarily used to automatically assign an IP addresses to internal network or Internet Service Provider (ISP), as a mean to give users or internal network administrators central management of all computers.



```
udp.port==138
No.   Time        Source        Destination       Protocol  Length  Info
188   26.996382   169.254.0.88  169.254.255.255   BROWSER   228     Request Announcement USER-BQG6J46QGR
189   26.997524   169.254.0.88  169.254.255.255   BROWSER   243     Host Announcement USER-BQG6J46QGR, Wor
194   28.496177   169.254.0.88  169.254.255.255   BROWSER   228     Request Announcement USER-BQG6J46QGR
197   29.996194   169.254.0.88  169.254.255.255   BROWSER   228     Request Announcement USER-BQG6J46QGR
243   37.458337   10.205.10.38  10.205.10.63      BROWSER   228     Request Announcement USER-BQG6J46QGR
244   37.461891   10.205.10.38  10.205.10.63      BROWSER   243     Host Announcement USER-BQG6J46QGR, Wor

> Frame 188: 228 bytes on wire (1824 bits), 228 bytes captured (1824 bits) on interface 0
> Ethernet II, Src: IntelCor_bc:aa:61 (e4:f8:9c:bc:aa:61), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 169.254.0.88, Dst: 169.254.255.255
> User Datagram Protocol, Src Port: 138, Dst Port: 138
> NetBIOS Datagram Service
> SMB (Server Message Block Protocol)
> SMB MailSlot Protocol
> Microsoft Windows Browser Protocol
```

Figure 4. UDP port 138

Port 138 Description: Both Ports 137 and 138 belong to the UDP port and transfer file information to each other on the LAN. 138 port is to provide the computer name browsing under the NetBIOS environment.



```
udp.port == 137
No.    Time        Source         Destination      Protocol  Length Info
   123 20.997155   169.254.0.88   169.254.255.255  NBNS      110 Registration NB USER-BQG6J46QGR<00>
   124 20.997959   169.254.0.88   169.254.255.255  NBNS      110 Registration NB WORKGROUP<00>
   125 20.997960   169.254.0.88   169.254.255.255  NBNS      110 Registration NB USER-BQG6J46QGR<20>
   140 21.746121   169.254.0.88   169.254.255.255  NBNS      110 Registration NB USER-BQG6J46QGR<20>
   141 21.746800   169.254.0.88   169.254.255.255  NBNS      110 Registration NB WORKGROUP<00>

> Frame 123: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) on interface 0
> Ethernet II, Src: IntelCor_bc:aa:61 (e4:f8:9c:bc:aa:61), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 169.254.0.88, Dst: 169.254.255.255
> User Datagram Protocol, Src Port: 137, Dst Port: 137
> NetBIOS Name Service
```

Figure 5. UDP port 137

Port 137 Description: Used primarily for NetBIOS Name Service. The user only needs to send a request to port 137 of a computer on the local area network or the Internet, then it can obtain the name of the computer and the registered user name and other information.

3) UDP Application Instance

The UDP protocol is applicable to scenarios that require high transmission efficiency and relatively low transmission accuracy. In the poor network environment, UDP protocol tends to have more packet loss. However, since UDP is not a connection-type protocol, it has the advantages of small resource consumption and high processing speed. So usually audio, video and ordinary data are transmitted using UDP, because even if one or two data packets are lost occasionally, it will not have much impact on the overall reception result of such data. At the same time UDP's high transmission efficiency also meets real-time transmission needs. For example, the social application QQ uses the UDP protocol[5].

TABLE 1. APPLICATIONS IN UDP

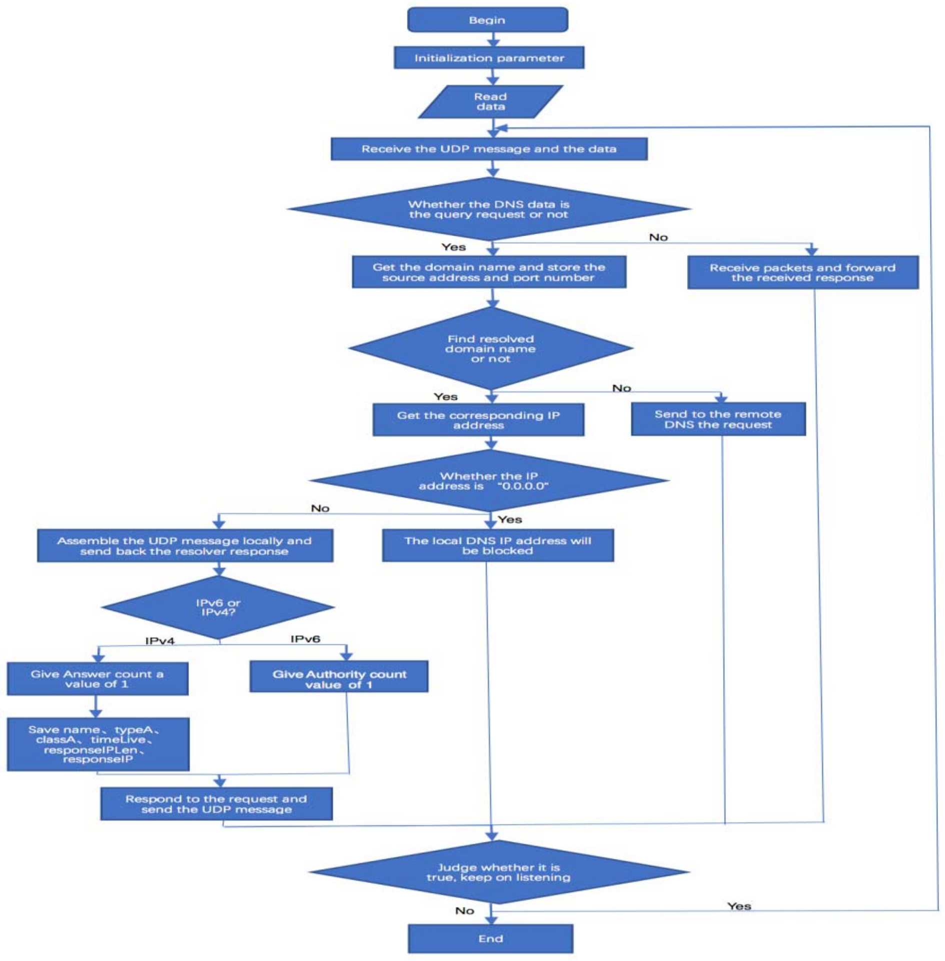
Application layer protocol	**application**
DNS	domain name service
TFTP	file transfer
RIP	routing protocol
SNMP	network management
NFS	remote file server
BOOTH，DHCP	IP address configuration

III. IMPLEMENTATION OF TRADITIONAL DNS

A Design Implementation

Binding the socket to the port whose port number is 53 and continuously take the action of listening. During the continuously listening, it will receive the UDP message and the data of DNS. Judging whether the DNS data is the query request or not. If the DNS data is not the query request, it will receive packets and forward the received response from the remote DNS. If the DNS data is the query request, it will get the domain name and store the source address and port number of the message. And then, the program will search the resolved domain name in the local domain name resolution table[6] .If the program can not find the resolved domain name, it will send to the remote DNS request and store this ID. If successfully find the resolved domain name, it will get the corresponding IP address and begin to judge whether the local DNS IP address is “0.0.0.0” or not. If the DNS IP address is “0.0.0.0”, the local DNS IP address will be blocked, the position of the cursor will be changed and it will wrap the data and send the data. If the local DNS IP address is not “0.0.0.0”, it will assemble the UDP message locally and send back the resolver response. And then judge whether the request is sent by IPv6 or IPv4[7]. If it is IPv6 request, change the position of the cursor and give Authority count a value of 1[8]. If it is IPv4 request, it will be returned by the package, the position of the cursor will be changed and it will give Answer count a value of 1[9]. And, the program will store name, typeA, classA, timeLive, responseIPLen and responseIP. After that the program will respond to the request and send the UDP message. At last, it begins to run and read the data.

B Flow Chart



Begin
Initialization parameter
Read data
Receive the UDP message and the data
Whether the DNS data is the query request or not
Yes
No
Get the domain name and store the source address and port number
Receive packets and forward the received response
Find resolved domain name or not
Yes
No
Get the corresponding IP address
Send to the remote DNS the request
Whether the IP address is "0.0.0.0"
No
Yes
Assemble the UDP message locally and send back the resolver response
The local DNS IP address will be blocked
IPv6 or IPv4?
IPv4
IPv6
Give Answer count a value of 1
Give Authority count value of 1
Save name、typeA、classA、timeLive、responseIPLen、responseIP
Respond to the request and send the UDP message
Judge whether it is true, keep on listening
No
Yes
End

C Pseudocode

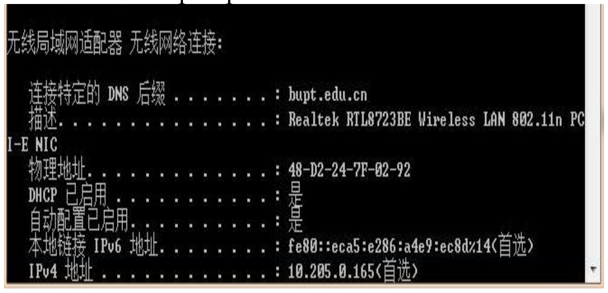
```
public void init() {
DatagramSocket socket ← null
try {
// bind to port 53
socket ← new DatagramSocket(LOCAL_PORT)
// keep on listening
while (true) {
// receive UDP message
socket.receive(inPacket)
// receive DNS data
byte[] sendData ← inPacket.getData()
// judge whether the DNS data is the query request or not
if (((sendData[2] & 0x80) == 0x00)) { // query
print ("\n Receive time：  " + new java.util.Date())
// get the domain name
domainNameStr ← getDomainName(sendData)
System.out.println("Domain name : " + domainNameStr)
// store the source address and port number of the message
resolverAddress ← inPacket.getAddress()
resolverPort ← inPacket.getPort()
// find the resolved domain name in the local domain name resolution table
if (Check.ipTable.containsKey(domainNameStr)) then {// get the
corresponding IP address
  String LocalDNSipAddress ← Check.ipTable.get(domainNameStr)
if (LocalDNSipAddress.equals("0.0.0.0"))  then {// If IP is 0.0.0.0
print ("Function：  " + "block")  // block
// change the position of the cursor response (flag=0x8183)
sendData[2] ← (byte) (sendData[2] | 0x81)
sendData[3] ← (byte) (sendData[3] | 0x83)
// wrap the data and send the data
outPacket←new DatagramPacket(sendData,sendData.length,
resolverAddress,resolverPort)
socket.send(outPacket)
IPv6_Flag ← false
} else {// If IP is not 0.0.0.0  assemble the UDP message locally and send
back the resolver response
byte[] finalData ← new byte[udpCursor + 16]
int cur ← 0  // answer cursor
if (IPv6_Flag) {// If it is IPv6 request
System.out.println("Function：" + "IPV6 locally response")
// change the position of the cursor response (flag=0x8180)
// give Authority count a value of 1
sendData[2] ← (byte) (sendData[2] | 0x81)
sendData[3] ← (byte) (sendData[3] | 0x80)
sendData[8] ← (byte) (sendData[8] | 0x00)
sendData[9] ← (byte) (sendData[9] | 0x01)
System.arraycopy(sendData, 0, finalData, cur,udpCursor)
IPv6_Flag ← false
} else {// IPv4 is returned by the package
System.out.println("Function：" + "IPV4 locally response")
// change the position of the cursor response (flag=0x8180)
// give Answer count a value of 1
sendData[2] ← (byte) (sendData[2] | 0x81)
sendData[3] ← (byte) (sendData[3] | 0x80)
sendData[6] ← (byte) (sendData[6] | 0x00)
sendData[7] ← (byte) (sendData[7] | 0x01)
System.arraycopy(sendData, 0, finalData, cur,udpCursor)
// store name
cur ← cur+udpCursor
byte[] name ← {(byte) 0xc0,(byte)0x0c}
System.arraycopy(name, 0,finalData, cur, 2)
// store typeA
cur ← cur + 2
byte[] typeA ← {(byte) 0x00,(byte)0x01}
System.arraycopy(typeA, 0,finalData, cur, 2)
// store classA
cur ← cur + 2
byte[] classA ← {(byte) 0x00,(byte)0x01}
System.arraycopy(classA, 0,finalData, cur, 2)
// store timeLive
cur ← cur + 2
byte[] timeLive ← {(byte)0x00,(byte)0x01,(byte)0x51,(byte)0x80}
System.arraycopy(timeLive, 0,finalData, cur, 4)
// store responseIPLen
cur ← cur + 4
byte[] responseIPLen ← {(byte) 0x00,(byte)0x04}
System.arraycopy(responseIPLen, 0,finalData, cur, 2)
// store responseIP
cur ← cur +  2
byte[] responseIP
←InetAddress.getByName( Check.ipTable.get(domainNameStr)).getAddre
ss()
System.arraycopy(responseIP, 0, finalData, cur,4)
cur ← cur + 4
} End if
// respond to the request and send the UDP message
outPacket ← new DatagramPacket(finalData,finalData.length,
resolverAddress,resolverPort)
socket.send(outPacket)
} End if
} else {// can not find the resolved domain name in the local domain name
resolution table
// send to the remote DNS request
outPacket ← new DatagramPacket(sendData,sendData.length,
InetAddress.getByName(DNS_IP),DNS_PORT)
socket.send(outPacket)
print("Forward time：  " + new java.util.Date())

IPv6_Flag ← false
print ("Function：" + "forward to the remote DNS")
// store this ID
IDTransition idTransition ← new IDTransition( (int) byte2Short(sendData,
0),resolverPort, resolverAddress)
idMap.put(idTransition.getOldID(), idTransition)
} End if
} else {// response
// receive packets
int responseID ← byte2Short(sendData, 0)
// outPacket = new DatagramPacket(sendData,
// sendData.length, resolverAddress, resolverPort)
// socket.send(outPacket)
if (idMap.containsKey(responseID)) then
{ IDTransition id ← idMap.get(responseID)
// forward the received response from the remote DNS
outPacket ← new DatagramPacket(sendData,sendData.length,
id.getAddr(), id.getPort())
```

```
socket.send(outPacket)
}
} End if
} End while
} catch (Exception e) {
// close socket
socket.close()
e.printStackTrace()
}
}
public static void main(String[] args) throws UnknownHostException {
try {
Check.readData("src\\dnsrelay.txt") // read data
} catch (IOException e) {
e.printStackTrace()
}
print("Name Server: " + DNS_IP)
print ("Debug Level: 0")
print ("Bind UDP port " + LOCAL_PORT + " ...OK!")
print ("Try to load table \"dnsrelay.txt\" " + " ...OK!")
print (Check.ipTable.size() + " names," + "occupy "+ Check.fileSize + "
bytes memory")
print
("========================================")
(new DNSRelay()).init() // start program
}
}
```

IV. Program Test

A Operation Result

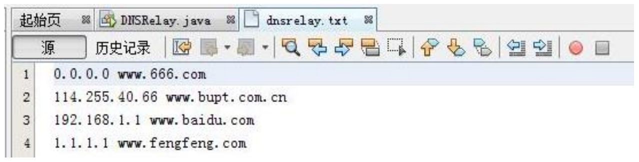
The test host is using a wireless network connection, wireless LAN adapter parameters are as follows



```
无线局域网适配器 无线网络连接:

   连接特定的 DNS 后缀 . . . . . . . : bupt.edu.cn
   描述. . . . . . . . . . . . . . . : Realtek RTL8723BE Wireless LAN 802.11n PC
I-E NIC
   物理地址. . . . . . . . . . . . . : 48-D2-24-7F-02-92
   DHCP 已启用 . . . . . . . . . . . : 是
   自动配置已启用. . . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . . : fe80::eca5:e286:a4e9:ec8d%14(首选)
   IPv4 地址 . . . . . . . . . . . . : 10.205.0.165(首选)
```

Figure 7. IP configuration

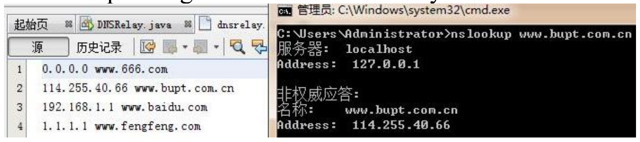
The local database presets the IP addresses of four domain names:



```
0.0.0.0 www.666.com
114.255.40.66 www.bupt.com.cn
192.168.1.1 www.baidu.com
1.1.1.1 www.fengfeng.com
```

Figure 8. Local database

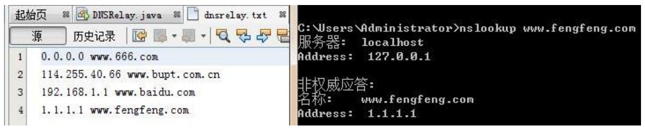
Then use CMD to run the program: Firstly, we test the domain name that are stored in local database. We received the corresponding IP address successfully.



```
0.0.0.0 www.666.com
114.255.40.66 www.bupt.com.cn
192.168.1.1 www.baidu.com
1.1.1.1 www.fengfeng.com
```

```
C:\Users\Administrator>nslookup www.bupt.com.cn
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    www.bupt.com.cn
Address:  114.255.40.66
```

Figure 9. Flow1

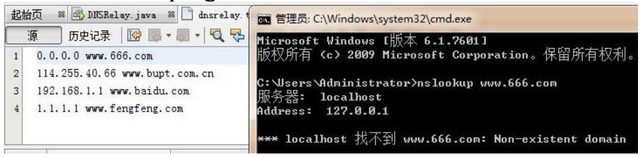


```
0.0.0.0 www.666.com
114.255.40.66 www.bupt.com.cn
192.168.1.1 www.baidu.com
1.1.1.1 www.fengfeng.com
```

```
C:\Users\Administrator>nslookup www.fengfeng.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    www.fengfeng.com
Address:  1.1.1.1
```

Figure 10. Flow2

In addition, if we enter the domain name that has the IP address 0.0.0.0, this local DNS IP address will be blocked. This means the program will return “Non-existent domain”.

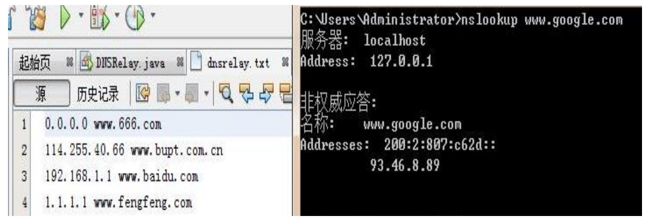


```
0.0.0.0 www.666.com
114.255.40.66 www.bupt.com.cn
192.168.1.1 www.baidu.com
1.1.1.1 www.fengfeng.com
```

```
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>nslookup www.666.com
服务器:  localhost
Address:  127.0.0.1

*** localhost 找不到 www.666.com: Non-existent domain
```

Figure 11. Flow3

Then we test the domain name that is not stored in the local database. So the program will access the DNS server to find the IP address.

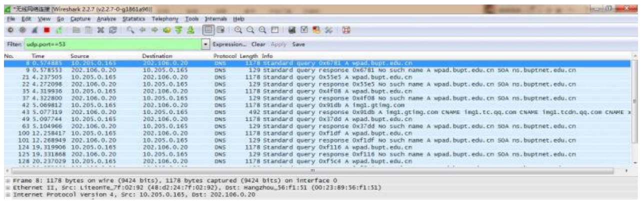


```
0.0.0.0 www.666.com
114.255.40.66 www.bupt.com.cn
192.168.1.1 www.baidu.com
1.1.1.1 www.fengfeng.com
```

```
C:\Users\Administrator>nslookup www.google.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    www.google.com
Addresses:  200:2:807:c62d::
          93.46.8.89
```

Figure 12. Flow4

We want to visit wpad.bupt.edu.cn which is not stored in the local database. The package we captured by Wireshark are shown as follows:

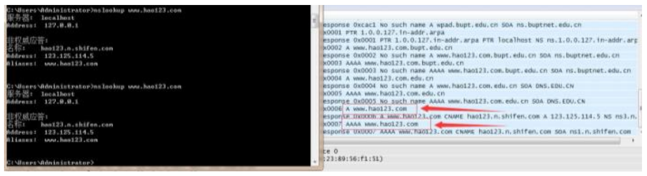


*无线网络连接 [Wireshark 2.2.7 (v2.2.7-0-g1861a96)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: udp.port==53
Expression... Clear Apply Save

```
No.  Time       Source        Destination   Protocol Length Info
  8 0.574885   10.205.0.165  202.106.0.20  DNS     1178 Standard query 0x6781 A wpad.bupt.edu.cn
  9 0.578553   202.106.0.20  10.205.0.165  DNS      129 Standard query response 0x6781 No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
 21 4.237505   10.205.0.165  202.106.0.20  DNS     1178 Standard query 0x55e5 A wpad.bupt.edu.cn
 22 4.272098   202.106.0.20  10.205.0.165  DNS      129 Standard query response 0x55e5 No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
 35 4.319936   10.205.0.165  202.106.0.20  DNS     1178 Standard query 0x4f08 A wpad.bupt.edu.cn
 37 4.322800   202.106.0.20  10.205.0.165  DNS      129 Standard query response 0x4f08 No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
 42 5.069812   10.205.0.165  202.106.0.20  DNS     1178 Standard query 0x91db A img1.gtimg.com
 43 5.077319   202.106.0.20  10.205.0.165  DNS      492 Standard query response 0x91db A img1.gtimg.com CNAME img1.tc.qq.com CNAME img1.tcdn.qq.com CNAME
 49 5.097744   10.205.0.165  202.106.0.20  DNS     1178 Standard query 0x37dd A wpad.bupt.edu.cn
 63 5.104966   202.106.0.20  10.205.0.165  DNS      129 Standard query response 0x37dd No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
100 12.258417  10.205.0.165  202.106.0.20  DNS     1178 Standard query 0xf1df A wpad.bupt.edu.cn
101 12.268949  202.106.0.20  10.205.0.165  DNS      129 Standard query response 0xf1df No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
124 19.319906  10.205.0.165  202.106.0.20  DNS     1178 Standard query 0xf116 A wpad.bupt.edu.cn
125 19.331868  202.106.0.20  10.205.0.165  DNS      129 Standard query response 0xf116 No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
128 20.237029  10.205.0.165  202.106.0.20  DNS     1178 Standard query 0xf5c4 A wpad.bupt.edu.cn

Frame 8: 1178 bytes on wire (9424 bits), 1178 bytes captured (9424 bits) on interface 0
Ethernet II, Src: LiteonTe_7f:02:92 (48:d2:24:7f:02:92), Dst: Hangzhou_56:f1:51 (00:23:89:56:f1:51)
Internet Protocol Version 4, Src: 10.205.0.165, Dst: 202.106.0.20
```

Figure 13. Flow5

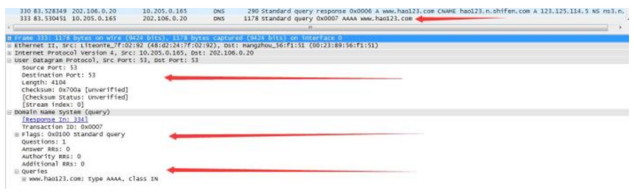
And there were the packages captured when we entered www.hao123.com. The data of OSI application layer, transport layer and network layer including the port numbers are shown explicitly.



```
C:\Users\Administrator>nslookup www.hao123.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    hao123.n.shifen.com
Address:  123.125.114.5
Aliases:  www.hao123.com


C:\Users\Administrator>nslookup www.hao123.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    hao123.n.shifen.com
Address:  123.125.114.5
Aliases:  www.hao123.com


C:\Users\Administrator>
```

```
esponse 0xcac1 No such name A wpad.bupt.edu.cn SOA ns.buptnet.edu.cn
x0001 PTR 1.0.0.127.in-addr.arpa
esponse 0x0001 PTR 1.0.0.127.in-addr.arpa PTR localhost NS ns.1.0.0.127.in-addr.arp
x0002 A www.hao123.com.bupt.edu.cn
esponse 0x0002 No such name A www.hao123.com.bupt.edu.cn SOA ns.buptnet.edu.cn
x0003 AAAA www.hao123.com.bupt.edu.cn
esponse 0x0003 No such name AAAA www.hao123.com.bupt.edu.cn SOA ns.buptnet.edu.cn
x0004 A www.hao123.com.edu.cn
esponse 0x0004 No such name A www.hao123.com.edu.cn SOA DNS.EDU.CN
x0005 AAAA www.hao123.com.edu.cn
esponse 0x0005 No such name AAAA www.hao123.com.edu.cn SOA DNS.EDU.CN
x0006 A www.hao123.com
esponse 0x0006 A www.hao123.com CNAME hao123.n.shifen.com A 123.125.114.5 NS ns3.n.
x0007 AAAA www.hao123.com
esponse 0x0007 AAAA www.hao123.com CNAME hao123.n.shifen.com SOA ns1.n.shifen.com

ce 0
:23:89:56:f1:51)
```

Figure 14. Flow6



```
330 83.528349  202.106.0.20    10.205.0.165    DNS    290 Standard query response 0x0006 A www.hao123.com CNAME hao123.n.shifen.com A 123.125.114.5 NS ns3.n.
333 83.530451  10.205.0.165    202.106.0.20    DNS   1178 Standard query 0x0007 AAAA www.hao123.com

⊞ Frame 333: 1178 bytes on wire (9424 bits), 1178 bytes captured (9424 bits) on interface 0
⊞ Ethernet II, Src: LiteonTe_7f:02:92 (48:d2:24:7f:02:92), Dst: Hangzhou_56:f1:51 (00:23:89:56:f1:51)
⊞ Internet Protocol Version 4, Src: 10.205.0.165, Dst: 202.106.0.20
⊟ User Datagram Protocol, Src Port: 53, Dst Port: 53
    Source Port: 53
    Destination Port: 53
    Length: 4104
    Checksum: 0x700a [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
⊟ Domain Name System (query)
    [Response In: 334]
    Transaction ID: 0x0007
  ⊞ Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊞ www.hao123.com: type AAAA, class IN
```

Figure 15. Flow7

V. SUMMARY

By Wireshark and the results, we prove that our program queried the local database preferentially. And return the corresponding IP address if there has. Or a DNS query message will be generated and sent to the network through UDP port 53. Accessing local database is more conveniently than querying the DNS server. Since the latter usually depends on the network environment. You may even can not receive the respond from the DNS server if there exists a timeout caused by congestion. Hence, our design improves the efficiency that find the IP address of a domain name.

The program can abandon the IPv6 and just use the IPv4 so that it can significantly increase the rate of query and access. It can also reduce the cache redundancy (in the underlying network) in the cache step of the query results.[10] And the proposal to this method is to improve the safety of the DNS message. In order to improve the safety of the query, the program should encrypt data at the network layer and verify the segment of the entire IP message.[11] Python can be used to crawl secure internet address, store the data in the database and classify the domain names according to the security classification (special domain, organizational domain, ordinary domain and so on).

REFERENCES

[1] Bellis R. DNS Transport over TCP - Implementation Requirements[J]. Poultry Science, 2010, 75(11):-.

[2] Eisink J H. ADDRESS AND PORT NUMBER ABSTRACTION WHEN SETTING UP A CONNECTION BETWEEN AT LEAST TWO COMPUTATIONAL DEVICES: US, US 20070168551 A1[P]. 2007.

[3] Jones E. DNS based enforcement for confinement and detection of network malicious activities: US, US7984493[P]. 2011.

[4] Weinberger J, Huitema C, Mahy R. STUN—Simple traversal of user datagram protocol (UDP) through network address translators (NATs)[C]// IEEE International Conference on Computer & Information Technology. IEEE Computer Society, 2003:22-27.

[5] Kalavade A P, Thomas J J, Palmer L A, et al. Domain name resolution:, US9426049[P]. 2016.

[6] Mao Z M, Cranor C D, Douglis F, et al. A Precise and Efficient Evaluation of the Proximity between Web Clients and their Local DNS Servers[C]// Usenix Technical Conference. 2008:229--242.

[7] Perkins C, Johnson D. Mobility support in ipv6 (work in progress) [J]. Rfc, 2003, 25(2): F2H-1-F2H-2.

[8] [8] Montenegro G, Kushalnagar N, Hui J, et al. RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks[J]. 2007.

[9] Cheshire S, Aboba B. Dynamic Configuration of IPv4 Link-Local Addresses[J]. Heise Zeitschriften Verlag, 2005((2).

[10] Liu, Zaide "Edward, Swildens, Eric Sven-johan, Day, Richard David. Domain name resolution using a distributed DNS network: US, US7725602[P]. 2010.

[11] Deccio C, Chen C C, Mohapatra P, et al. Quality of name resolution in the Domain Name System[C]// IEEE International Conference on Network Protocols. IEEE, 2009:113-122.

[12] Linington P F. Open system interconnection (OSI)[C]// John Wiley and Sons Ltd. 2003:1288-1289.

[13] Mehta R, Bhandari S, Bansal S, et al. Studying the Open System Interconnection Model and Proposing the Concept of Layer Zero[J]. Indian Journal of Science & Technology, 2016, 9(21).

[14] Ruiz-SáNchez M A, Biersack E W, Dabbous W. Survey and taxonomy of IP address lookup algorithms[J]. IEEE Network, 2001, 15(2):8-23.

[15] Xie Y, Yu F, Achan K, et al. How dynamic are IP addresses?[J]. Acm Sigcomm Computer Communication Review, 2007, 37(4):301-312.

[16] Hotz S M, Joffe R L, Manning W C, et al. Domain name resolution system and method: US, US20040039798[P]. 2004.

[17] Jackson J, Sheth N S. System for communicating with an internet protocol multimedia subsystem network: US, US9154526[P]. 2015.

[18] Supriyanto, Hasbullah I H, Murugesan R K, et al. Survey of Internet Protocol Version 6 Link Local Communication Security Vulnerability and Mitigation Methods[J]. Iete Technical Review, 2013, 30(1):64-71.

[19] Qian H, Rabinovich M, Al-Qudahj Z. Bringing Local DNS Servers Close to Their Clients[C]// Global Telecommunications Conference. IEEE, 2013:1-6.