

Undergraduate Lab Report of Jinan University

Course Title Computer Networks Lab Evaluation
Lab Name DNS: Domain Name System
Lab Address N117 Instructor Dr. CUI Lin (崔林)
Student Name 蒋云翔 Student No 2022102330
College International School
Department Major CST
Date 2024 / 12 / 3

1. Introduction

1) Objective

- Know the format of DNS message.
- Understand architecture and operations of DNS.
- Understand the usage of DNS cache.

2) Experiment Principle

1. Introduction of DNS

The Domain Name System (DNS) associate's domain names with corresponding addresses and vice versa, enabling users to interact with websites through familiar domain names. As a decentralized database and a client/server application, DNS provides robust fault tolerance due to its distributed architecture.

2. Domain Name Space

The design of the domain name space is hierarchical, resembling an inverted tree with the root at the apex. This structure is composed of up to 128 levels, ranging from level 0, which is the root, down to level 127.

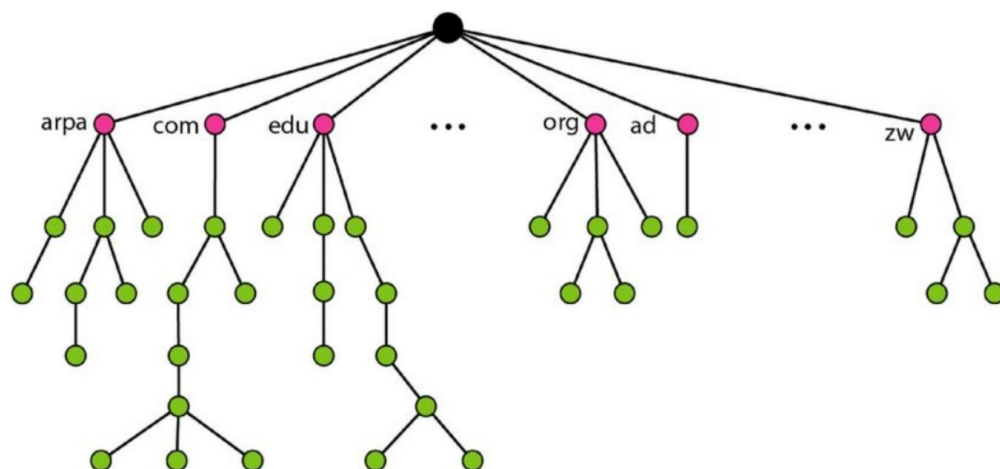


Figure 1: Domain Name Space

2.1. Label

In the domain name system structure, every node within the tree hierarchy is assigned a label that can be up to 63 characters long. The root of this tree is denoted by an empty string, effectively a null label. To ensure the distinctiveness of each domain name across the network, DNS mandates that sibling nodes, namely those emanating from the same parent node each have unique labels.

2.2. Domain Name

Each node in the domain name system hierarchy represents a domain name, which consists of a series of labels separated by dots (".") and read from the node to the root. The terminating label of a fully qualified domain name is an empty string corresponding to the root node, which is why a complete domain name ends with a dot, signifying the null label of the root.

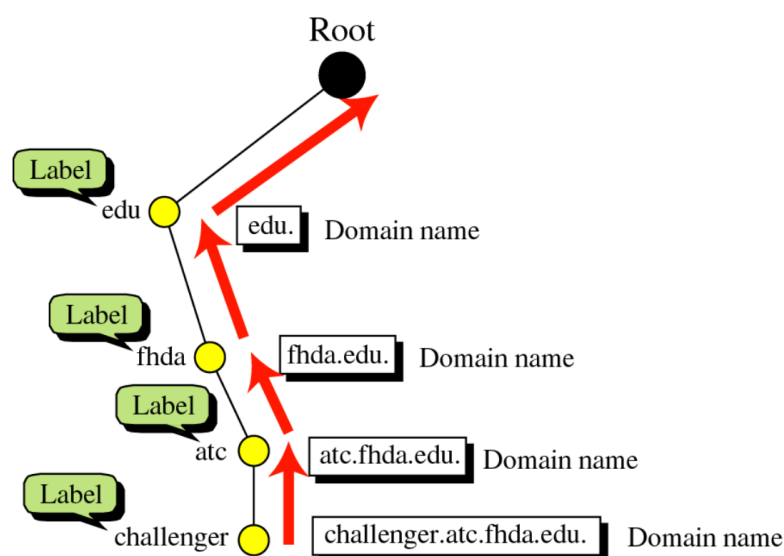


Figure 2: Domain names and labels

2.2.1. Fully Qualified Domain Name

A fully qualified domain name (**FQDN**) is a domain name that specifies its exact location in the hierarchy of the Domain Name System. It includes every individual label from the host-specific to the top-level domain, thus completely and uniquely identifying the host's position within the DNS tree. The FQDN ends with a dot that represents the root, though it is often omitted. For instance, 'challenger.atc.fhda.edu' is an FQDN for a host called 'challenger', specifying its full path within the DNS hierarchy.

2.2.2. Partially Qualified Domain Name

A partially qualified domain name (PQDN) lacks termination by a null string, indicating it does not extend to the root of the DNS hierarchy. A PQDN begins at a certain node and implies a local context, assuming the remaining path within a familiar domain. The resolver appends the requisite suffix based on the local domain to construct a full FQDN. For instance, within the 'atc.fhda.edu' domain, a user can

simply use 'challenger' to refer to the host, and the DNS client will concatenate the local domain, forming 'challenger.atc.fhda.edu', to query the DNS server.

2.2.3. Domain

A domain represents a sub-section of the domain name space, taking the form of a subtree. The domain's name is taken from the domain name assigned to the highest node within that subtree. As illustrated in the following figure, domains can be further subdivided into smaller domains, often referred to as subdomains.

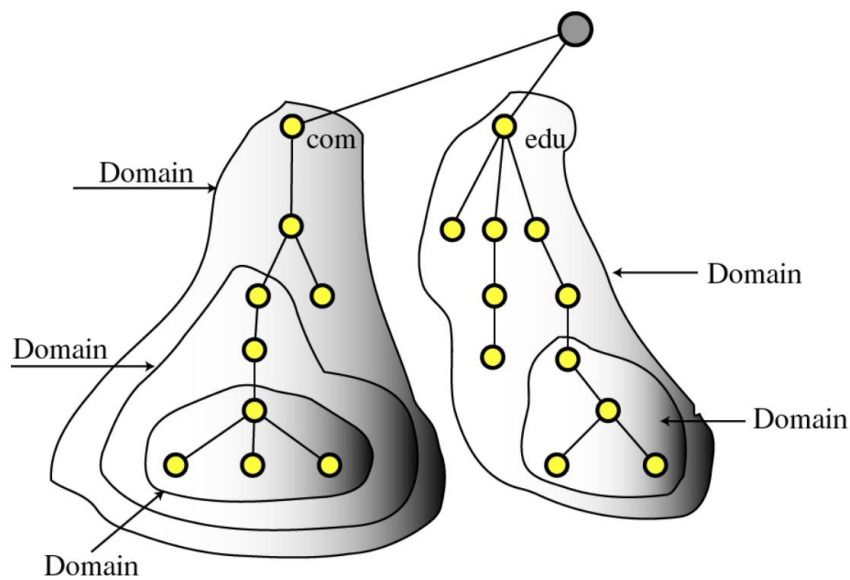


Figure 3: Domain

3. Distribution of Name Space

On the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain.

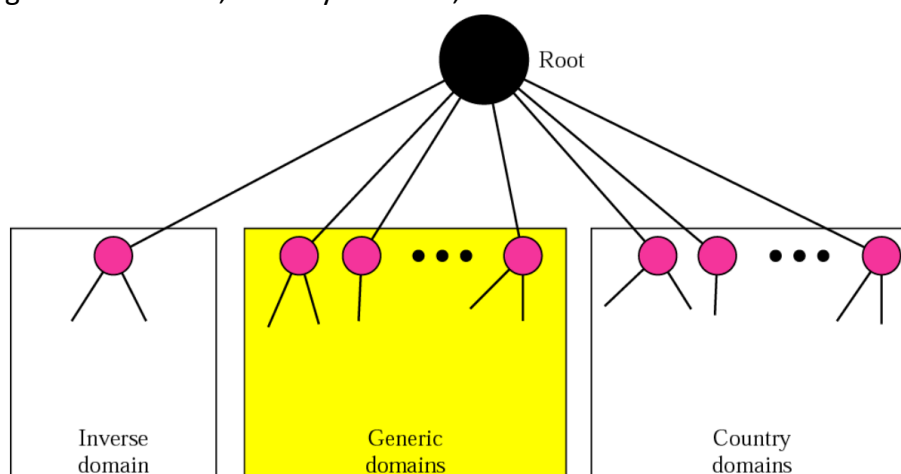


Figure 4: DNS used on the Internet

3.1. Generic Domains

Generic domains categorize registered hosts based on their general

characteristics. In the domain name space, each node signifies a domain, serving as an index within the domain name database

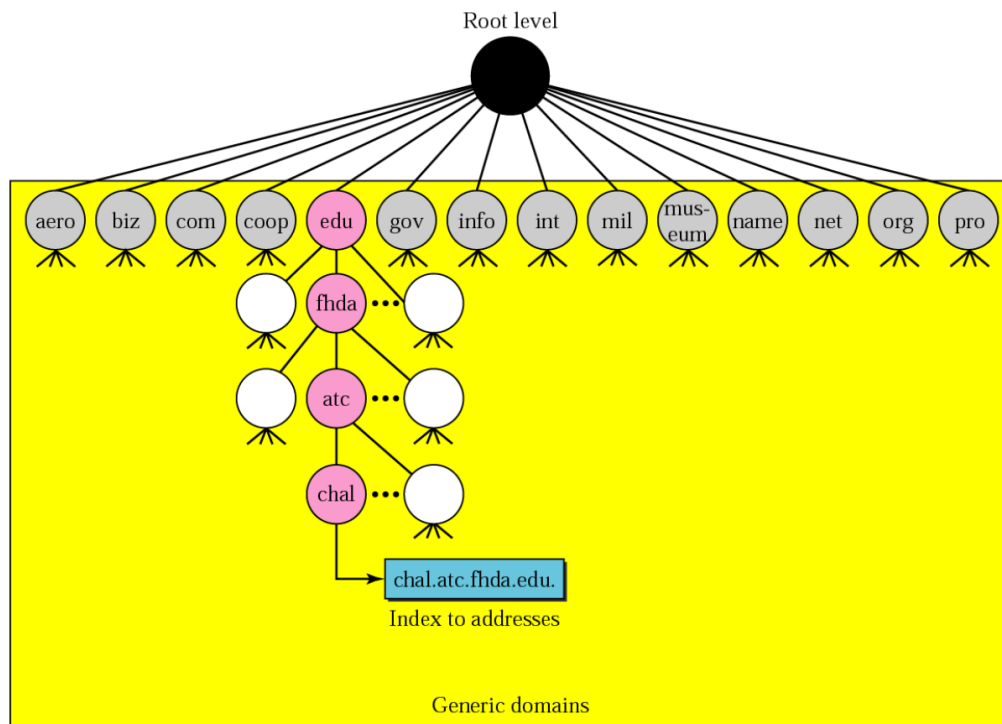


Figure 5: Generic domains

In the generic domains section of the DNS hierarchy, the first level encompasses 14 distinct labels. These labels categorize domains based on the type of organization they represent, as detailed in following table.

Label	Description	Label	Description
aero	Airlines and aerospace companies	int	International organizations
biz	Businesses or firms (similar to "com")	mil	Military groups
com	Commercial organizations	museum	Museums and other nonprofit organizations
coop	Cooperative business organizations	name	Personal names (individuals)
edu	Educational institutions	net	Network support centers
gov	Government institutions	org	Nonprofit organizations
info	Information service providers	pro	Professional individual organizations

Figure 6: Generic domain labels

3.2. Country Domains

In the country domains section of DNS, two-letter abbreviations are used to represent each country, like 'cn' for China. The second-level labels within these country domains may denote organizations or more specific national categorizations.

3.3. Inverse Domain

The inverse domain in DNS is utilized for mapping an IP address to a domain name, often when a server needs to verify if a client is authorized. It uses a special domain, with the top-level node named 'arpa' (for historical reasons) and a second-level node 'in-addr' (for inverse address). This domain is structured hierarchically, reflecting the components of an IP address in reverse order. For instance, the IP address 132.34.45.121 is represented as 121.45.34.132.in-addr.arpa. This setup, different from generic or country domains, allows for organizing the network ID higher than the subnet ID, and the subnet ID higher than the host ID, facilitating efficient management of requests to map addresses to names. An illustration of this structure is shown in Figure 7.

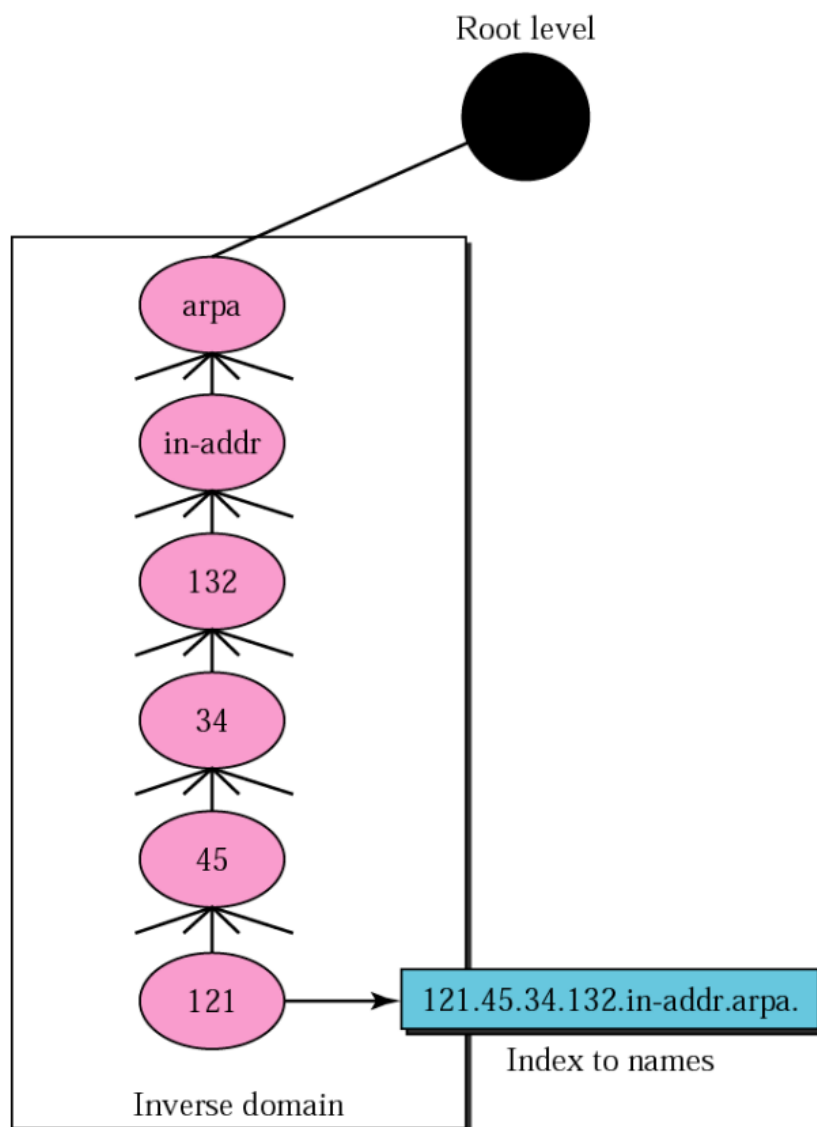


Figure 7: Inverse domain

4. Format of DNS message

DNS message consists of 12-Byte header and 4 variable fields:

Identification (16-bit)	Flags (16-bit)
Number of question records (16-bit)	Number of answer records (16-bit)
Number of authoritative records (16-bit)	Number of additional records (16-bit)
Question section	
Answer section (available number)	
Authoritative section (available number)	
Additional section (available number)	

Figure 8: DNS message format

- (1) **Identification**: A 2-Byte field used by clients to match responses with queries.
(2) **Flags**: A 2-Byte field comprising various subfields that define the message type:

QR	Opcode	AA	TC	RD	RA	Reversed	AD	CD	rcode
1-bit	4-bit	1-bit	1-bit	1-bit	1-bit	1-bit	1-bit	1-bit	4-bit

Figure 9: DNS Flags fields

The description of each field is shown as follows:

- **QR** (Query/Response): Set to 0 for queries and 1 for responses.
- **Opcode**: Indicates query types, such as 0 for standard, 1 for inverse, and 2 for status requests.
- **AA** (Authoritative Answer): Set to 1 to signify an authoritative server response.
- **TC** (Truncation): Indicates message truncation due to size exceeding transmission limits.
- **RD** (Recursion Desired): Set by a client to request recursive resolution.
- **RA** (Recursion Available): Indicates in responses if recursive query support is available.
- **AD** (Authenticated data): Shows data authenticity as confirmed by the server.
- **CD** (Checking Disabled): Indicates the client's willingness to accept unverified data.
- **Rcode**: A 4-bit field indicating any error in the DNS response.

- (3) **Number of question records:** Indicates the count of queries in the question section, stored in a 2-Byte field.
- (4) **Number of answer records:** Specifies the number of answer records in the response's answer section, also in a 2-Byte field.
- (5) **Number of authoritative records:** Holds the count of authoritative records in the response's authoritative section, using a 2-Byte field.
- (6) **Number of additional records:** Contains the number of additional records in the response's additional section, recorded in a 2-Byte field.
- (7) **Question Section:** This is a section consisting of one or more question records. It is present in both query and response messages. The format of each question is shown as follows:

Query name	
Query type	Query class

Figure 10: DNS question section

- **Query name:** The domain being queried.
- **Query type:** Type of resource records requested.
- **Query class:** Class of the resource records.

- (8) **Answer Section:** Consists of resource records in responses, containing

Domain name	
Type	Class
Time to live	
Resource data length	Resource data

Figure 11: DNS answer section

- **Domain name:** Formatted similarly to the Query Name.
- **Type:** Corresponding to the Query Type.
- **Class:** Matching the Query Class.
- **Time to Live:** TTL value in seconds.
- **Resource Data Length:** A 2-Byte field indicating resource data length.
- **Resource Data:** Data corresponding to the resource record type.

- (9) **Authoritative Section:** It is composed of one or more resource records and

appears only in response messages. This section provides details about authoritative servers relevant to the query, typically including their domain names.

Domain name	
Type	Class
Time to live	
Resource data length	Resource data

Figure 12: DNS authoritative section

- **Domain name:** Formatted similarly to the Query Name.
- **Type:** Corresponding to the Query Type.
- **Class:** Matching the Query Class.
- **Time to Live:** TTL value in seconds.
- **Resource Data Length:** A 2-Byte field indicating resource data length.
- **Resource Data:** Data corresponding to the resource record type.

5. Forward Lookup versus Reverse Lookup

5.1. Resolver

DNS operates on a client/server model. When a host requires an IP-to-name or name-to-IP mapping, it utilizes a DNS client known as a resolver. The resolver queries the nearest DNS server. If the server has the required information, it directly responds to the resolver. If not, it either points the resolver to other servers or contacts them itself to fetch the information. Upon receiving the mapping, the resolver interprets the response to determine if it's a valid resolution or an error and then conveys the outcome to the requesting process.

5.2. Forward Lookup: Mapping Names to Addresses

Mostly, a resolver requests a server for an IP address corresponding to a given domain name. The server then searches the generic or country domains for this mapping. When the domain name falls under generic domains, the resolver forwards the query to a local DNS server. If the local server can't resolve it, it either directs the resolver to other servers or queries them directly. Similarly, for domain names in the country domains section, the process is the same: the resolver starts with the local server and then may extend the query to other servers for resolution.

5.3. Reverse Lookup: Mapping Addresses to Names

When a client seeks to map an IP address to a domain name, it initiates a PTR (pointer) query. DNS handles these using the inverse domain. In the query, the IP address is reversed, followed by appending the labels 'in-addr' and 'arpa' to form a

domain name suitable for the inverse domain. For example, for the IP address 132.34.45.121, the resolver reverses it and adds labels, creating “121.45.34.132.in-addr. arpa.” This modified domain name is then sent to the local DNS server for resolution.

6. Recursive Resolution versus Iterative Resolution

6.1. Recursive Resolution

When a client (resolver) requests a recursive answer from a DNS server, it expects the server to provide a definitive response. If the queried server is authoritative for the domain, it directly checks its database and responds. If not, the server forwards the request to a higher-level server (usually the parent server) and awaits its reply. This process continues until an authoritative server is found and responds. The answer is then relayed back through the chain of servers to the original client. This method, known as recursive resolution, ensures that the client receives a complete response to its query.

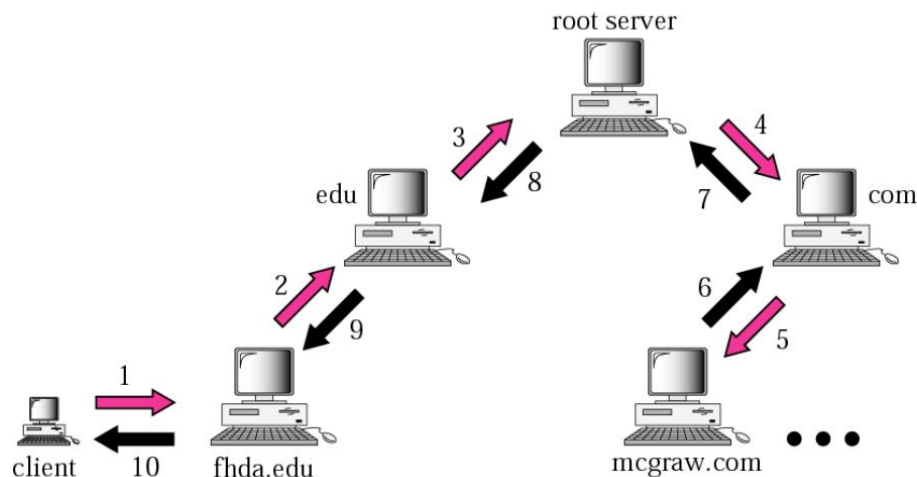


Figure 14: Recursive Resolution

6.2. Iterative Resolution

When a client doesn't request a recursive response, DNS resolution can proceed iteratively. In this process, if the server is authoritative for the queried name, it directly answers. If not, it provides the client with the IP address of a server it believes can fulfill the query. The client then contacts this next server. If this server can resolve the query, it responds with the necessary IP address. If it can't, it refers the client to another server. The client continues querying servers in this manner until it receives the answer. This method, known as iterative resolution, involves the client repeatedly querying different servers until it obtains the required information, as depicted in Figure 15 where the client consults four servers before receiving a response.

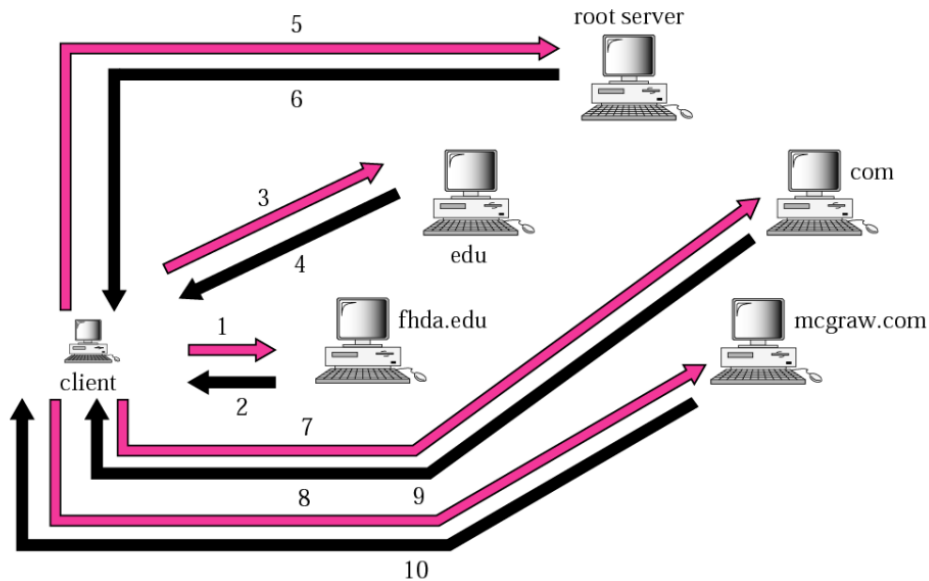


Figure 15: Iterative Resolution

7. Caching

DNS efficiency is enhanced through caching. When a server queries for a name outside its domain and receives a response from another server, it stores this response in cache memory. This allows for quicker responses to future queries for the same name, either from the same or different clients. However, cached responses are marked as non-authoritative to distinguish them from responses obtained directly from the authoritative source

To prevent outdated information due to prolonged caching, two methods are employed:

- (1) Authoritative servers include a Time-To-Live (TTL) value with each mapping, indicating how long it can be cached.
- (2) Servers maintain a TTL counter for each cached mapping and must purge entries with expired TTLs to ensure up-to-date information is provided to clients.

8. Compression

DNS messages often contain multiple domain names, especially in responses where these names are typically related or within the same zone. To optimize space, DNS uses message compression, which eliminates the need to repeat entire domain names.

When a DNS message includes a domain name that recurs, the subsequent appearances are replaced with a special pointer, a two-Byte subfield. This pointer uses the first two bits set to "11" (binary) and the remaining 14 bits to indicate the position in the message where the name first appears. The offset is measured from the start of the message, with the first Byte considered as position 0.

For instance, if the name "mail.xyzindustries.com" first appears at Byte 47, later references to this name in the same message are replaced with a pointer encoded as "11000000 00101111" (or in decimal, 192 and 47). This way, the second mention of "mail.xyzindustries.com" consumes only 2 Bytes instead of 24.

To distinguish between actual names and pointers, the design utilizes the value range. Labels in a domain name are restricted to 63 Bytes or less. Hence, if the first Byte of a name is 63 or less, it's interpreted as part of a real name. If it's 192 or higher, it's recognized as a pointer. This method ensures efficient use of space in DNS messages, especially when dealing with longer domain names or responses with multiple records.

9. Encapsulation

DNS can operate using either UDP or TCP, with both protocols utilizing port 53. Typically, UDP is preferred for responses under 512 Bytes, aligning with the common UDP packet size limit. For larger responses, DNS switches to TCP, which is more suitable for handling bigger data volumes without the constraint of the UDP packet size limit.

When a large response, such as a zone transfer between DNS servers, is expected, a TCP connection is directly established. If the response size is initially unknown, UDP may be used first. Should the response exceed 512 Bytes, the server truncates it and sets the TC (Truncation) flag, prompting the resolver to switch to TCP and reissue the request for the complete response.

2. Lab Environment

I accomplish this lab at home using wireless network (My Hotspot) (WLAN).



```
C:\WINDOWS\system32\ x + v
Microsoft Windows [版本 10.0.22631.4460]
(c) Microsoft Corporation。保留所有权利。

C:\Users\86136>hostname
YusiShrimp

C:\Users\86136>
```

Figure 16: Computer name

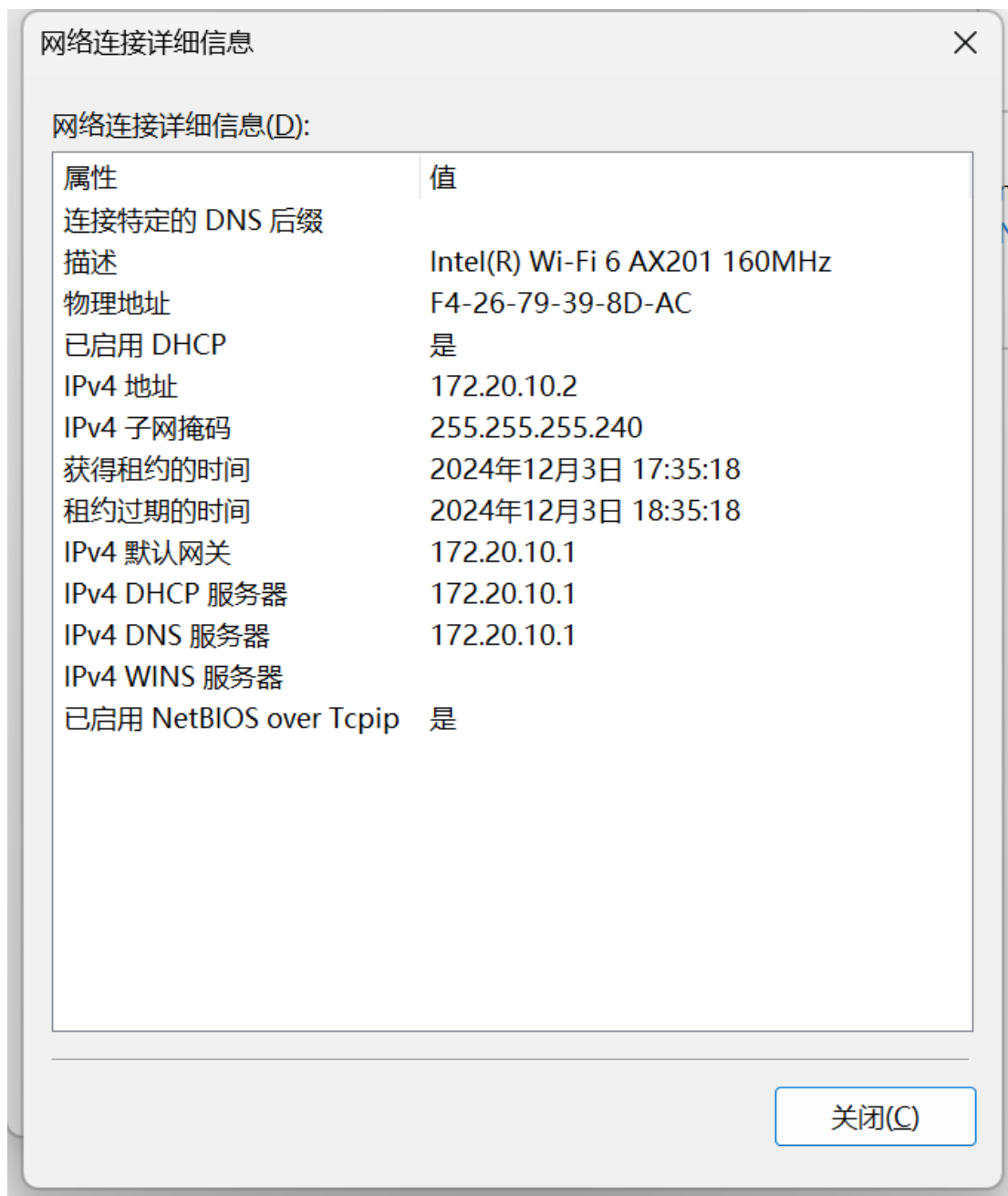


Figure 17: Detailed information of WLAN

3. Experiment Steps and Results

1) Task 1: Domain Name Resolution with nslookup

1. Experiment steps:

- [1] **Generic domain:** Run “nslookup www.python.org” command in the command line window.
- [2] **Country domain:** Run “nslookup www.gd.gov.cn” command.
- [3] **Reverse domain:** Run “nslookup 8.8.8.8” command.

2. Results:

The result of running “nslookup www.python.org”, “nslookup www.gd.gov.cn” and “nslookup 8.8.8.8” commands respectively are as follows:

```
C:\WINDOWS\system32\ x + v
Microsoft Windows [版本 10.0.22631.4460]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\86136>nslookup www.python.org
服务器: Unknown
Address: 192.168.0.1

非权威应答:
名称: dualstack.python.map.fastly.net
Addresses: 2a04:4e42:1a::223
           151.101.108.223
Aliases: www.python.org

C:\Users\86136>nslookup www.gd.gov.cn
服务器: Unknown
Address: 192.168.0.1

非权威应答:
名称: www.gd.gov.cn
Addresses: 2409:8754:2:1::d24c:4b59
           120.197.33.11
           210.76.75.89
           157.122.49.11

C:\Users\86136>nslookup 8.8.8.8
服务器: Unknown
Address: 192.168.0.1
```

Local name server (I use WLAN to complete this lab)

Corresponding correct IP addr (Both IPv4 and IPv6)

Some sites may have more than one IP address

Figure 18: Running “nslookup” to different domains name

3. Answer the questions

Q1: What is the corresponding IP address of “www.python.org”?

A1: The IPv4 address is “151.101.108.223”. There’s also an IPv6 address listed: “2a04:4e42:8c::223”.

Q2: What is the top-level domain for “www.python.org”?

A2: The top-level domain for “www.python.org” is “org”.

Q3: What is the corresponding IP address of “www.gd.gov.cn”?

A3: The IPv4 addresses are “120.197.33.11”, “210.76.75.89”, and “157.122.49.11”. There’s also an IPv6 address is “2409:8754:2:1::d24c:4b59”.

Q4: What is the top-level domain, second-level domain and third-level domain of “www.gd.gov.cn”?

A4: The top-level domain is “.cn”, then second level domain is “gov.cn” and the third level domain is “gd.gov.cn”.

Q5: What is the corresponding domain name of IP address 8.8.8.8?

A5: The corresponding domain name for the IP address 8.8.8.8 is “dns.google”.

Q6: What are the top-level and second-level domain names of reverse domain?

A6: For the reverse domain of IP address 8.8.8.8, the TLD is .google and the SLD is dns.

2) Task 2: Manually Name Resolution with dig

1. Experiment steps:

- [1] **Step 1:** Pick a domain name to resolve, such as www.jnu.edu.cn, www.gd.gov.cn or www.gznet.edu.cn. Find the IP address of one of the roots nameservers by searching on the web. You need this information to begin the name resolution process, and nameservers are provided with it as part of their configuration.
- [2] **Step 2:** Close all unnecessary browser tabs and applications. Browsing web sites will generate DNS traffic as your browser resolves domain names to connect to remote servers. We want to minimize browser activity initially so that we capture only the intended DNS traffic.
- [3] **Step 3:** Issue a request to one root nameserver to perform the first step of the resolution. The following example will use “A” root nameserver:

dig @198.41.0.4 www.gznet.edu.cn

The reply from the root nameserver does not provide the full name resolution, but it does tell us about nameservers closer to having the information for us to contact. In this case, it is nameservers who know about the “.cn” domain. Multiple nameservers are given as alternative choices, and the reply helpfully includes their IP addresses; We can see both IPv6 and IPv4 addresses.

- [4] **Step 4:** Now you have nameservers closer to having the answer. Issue a request to one of these name servers for further resolution.
- [5] **Step 5:** Continue the resolution process with dig until you complete the resolution, i.e., obtain the IP addresses of www.gznet.edu.cn.
- [6] **Step 6:** Now, launch Wireshark and start a capture with a filter of “dns” or “udp port 53”. Repeat the dig commands from the previous steps. This time, you should see the DNS request and reply packets that correspond to your commands captured in the trace window.
- [7] **Step 7:** Stop the Wireshark, save the trace and inspect the trace you captured (Check details of both DNS query and response messages).
- [8] **Step 8:** Draw a figure that shows the sequence of remote nameservers that you contacted and the domain for which they are responsible.
- [9] **Step 9:** You can also try to trace DNS path automatically using dig:

2. Results:

I selected the domain name www.gznet.edu.cn for manual domain name resolution experiment.

```
C:\WINDOWS\system32\ x + v
C:\Users\86136>dig @198.41.0.4 www.gznet.edu.cn

;<<>> DiG 9.4.3 <<>> @198.41.0.4 www.gznet.edu.cn
; (1 server found)
; global options: printcmd
; Got answer:
;-->HEADER<-- opcode: QUERY, status: NOERROR, id: 836
; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 11
; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.gznet.edu.cn.      IN      A

;; AUTHORITY SECTION:
cn.      172800 IN      NS      e.dns.cn.
cn.      172800 IN      NS      a.dns.cn.
cn.      172800 IN      NS      c.dns.cn.
cn.      172800 IN      NS      ns.cernet.net.
cn.      172800 IN      NS      d.dns.cn.
cn.      172800 IN      NS      b.dns.cn.

;; ADDITIONAL SECTION:
e.dns.cn. 172800 IN      A      203.119.29.1
e.dns.cn. 172800 IN      AAAA   2001:dc7:3::1
a.dns.cn. 172800 IN      A      203.119.25.1
a.dns.cn. 172800 IN      AAAA   2001:dc7::1
c.dns.cn. 172800 IN      A      203.119.27.1
c.dns.cn. 172800 IN      AAAA   2001:dc7:2::1
ns.cernet.net. 172800 IN      A      202.112.0.44
d.dns.cn. 172800 IN      A      203.119.28.1
d.dns.cn. 172800 IN      AAAA   2001:dc7:1000::1
b.dns.cn. 172800 IN      A      203.119.26.1
b.dns.cn. 172800 IN      AAAA   2001:dc7:1::1

;; Query time: 225 msec
;; SERVER: 198.41.0.4#53(198.41.0.4)
;; WHEN: Thu Dec 05 14:41:53 2024
;; MSG SIZE rcvd: 381

C:\Users\86136>
```

Figure 19: Using dig command to get the DNS server address of “cn.”

Each time I selected the address of the first DNS server provided by “Additional” section for the next step of domain name resolution. Finally, I successfully found the IPv4 address corresponding to this domain name through 4 dig command operations as “202.38.251.178”

```
C:\WINDOWS\system32\ x + v
;; Query time: 225 msec
;; SERVER: 198.41.0.4#53(198.41.0.4)
;; WHEN: Thu Dec 05 14:41:53 2024
;; MSG SIZE rcvd: 381

C:\Users\86136>dig @203.119.29.1 www.gznet.edu.cn

;<<>> DiG 9.4.3 <<>> @203.119.29.1 www.gznet.edu.cn
; (1 server found)
; global options: printcmd
; Got answer:
;-->HEADER<-- opcode: QUERY, status: NOERROR, id: 2045
; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 6
; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.gznet.edu.cn.      IN      A

;; AUTHORITY SECTION:
edu.cn. 172800 IN      NS      dns.edu.cn.
edu.cn. 172800 IN      NS      ns2.cernet.net.
edu.cn. 172800 IN      NS      ns4.cernet.net.
edu.cn. 172800 IN      NS      ns5.cernet.net.
edu.cn. 172800 IN      NS      dns2.edu.cn.
edu.cn. 172800 IN      NS      dns3.edu.cn.

;; ADDITIONAL SECTION:
dns.edu.cn. 172800 IN      A      202.38.109.35
dns.edu.cn. 172800 IN      AAAA   2001:250:c006::35
dns2.edu.cn. 172800 IN      A      202.112.0.13
dns2.edu.cn. 172800 IN      AAAA   2001:da8:1:100::13
dns3.edu.cn. 172800 IN      A      101.4.62.35
dns3.edu.cn. 172800 IN      AAAA   2001:250:62::35

;; Query time: 12 msec
;; SERVER: 203.119.29.1#53(203.119.29.1)
;; WHEN: Thu Dec 05 14:40:25 2024
;; MSG SIZE rcvd: 324

C:\Users\86136>
```

Figure 20: Using dig command to get the DNS server address of “edu.cn.”

```
C:\WINDOWS\system32\ x + -
dns3.edu.cn. 172800 IN A 101.4.62.35
dns3.edu.cn. 172800 IN AAAA 2001:250:62::35

;; Query time: 12 msec
;; SERVER: 203.119.29.1#53(203.119.29.1)
;; WHEN: Thu Dec 05 14:48:25 2024
;; MSG SIZE rcvd: 324

C:\Users\86136>dig @202.38.109.35 www.gznet.edu.cn

<<>> DiG 9.4.3 <<>> @202.38.109.35 www.gznet.edu.cn
(1 server found)
;; global options: printcmd
;; Got answer:
;; -->HEADER<-- opcode: QUERY, status: NOERROR, id: 905
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 6
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.gznet.edu.cn. IN A

;; AUTHORITY SECTION:
gznet.edu.cn. 172800 IN NS emily.scut.edu.cn.
gznet.edu.cn. 172800 IN NS scutsv33.scut.edu.cn.
gznet.edu.cn. 172800 IN NS orange.gznet.edu.cn.

;; ADDITIONAL SECTION:
scutsv33.scut.edu.cn. 172800 IN A 202.38.193.33
orange.gznet.edu.cn. 172800 IN A 202.112.17.33
emily.scut.edu.cn. 172800 IN A 202.38.251.33
scutsv33.scut.edu.cn. 172800 IN AAAA 2001:da8:2000:2193::33
orange.gznet.edu.cn. 172800 IN AAAA 2001:da8:2000:2017::33
emily.scut.edu.cn. 172800 IN AAAA 2001:da8:2000:2251::33

;; Query time: 49 msec
;; SERVER: 202.38.109.35#53(202.38.109.35)
;; WHEN: Thu Dec 05 14:50:05 2024
;; MSG SIZE rcvd: 241

C:\Users\86136>
```

Figure 21: Using dig command to get the DNS server address of “gznet.edu.cn.”

```
gznet.edu.cn. 172800 IN NS scutsv33.scut.edu.cn.
gznet.edu.cn. 172800 IN NS orange.gznet.edu.cn.

;; ADDITIONAL SECTION:
scutsv33.scut.edu.cn. 172800 IN A 202.38.193.33
orange.gznet.edu.cn. 172800 IN A 202.112.17.33
emily.scut.edu.cn. 172800 IN A 202.38.251.33
scutsv33.scut.edu.cn. 172800 IN AAAA 2001:da8:2000:2193::33
orange.gznet.edu.cn. 172800 IN AAAA 2001:da8:2000:2017::33
emily.scut.edu.cn. 172800 IN AAAA 2001:da8:2000:2251::33

;; Query time: 49 msec
;; SERVER: 202.38.109.35#53(202.38.109.35)
;; WHEN: Thu Dec 05 14:50:05 2024
;; MSG SIZE rcvd: 241

C:\Users\86136>dig @202.38.193.33 www.gznet.edu.cn

<<>> DiG 9.4.3 <<>> @202.38.193.33 www.gznet.edu.cn
(1 server found)
;; global options: printcmd
;; Got answer:
;; -->HEADER<-- opcode: QUERY, status: NOERROR, id: 244
;; flags: qr aa rd; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.gznet.edu.cn. IN A

;; ANSWER SECTION:
www.gznet.edu.cn. 86400 IN CNAME www2.scut.edu.cn.
www2.scut.edu.cn. 600 IN CNAME unifiweb.scut.edu.cn.
unifiweb.scut.edu.cn. 600 IN A 202.38.251.178

;; Query time: 18 msec
;; SERVER: 202.38.193.33#53(202.38.193.33)
;; WHEN: Thu Dec 05 14:52:16 2024
;; MSG SIZE rcvd: 97

C:\Users\86136>
```

We finally get the IP address of “www.gznet.edu.cn”, which is 202.38.251.178

Figure 22: Using dig command to get the DNS server address of www.gznet.edu.cn.

Repeating the above process, I use Wireshark to capture packets and set up an IP address filter to search for a DNS server closer to the domain name www.gznet.edu.cn each time. I can get the DNS requests and responses of the entire manual domain name resolution process. The capture content is as follows:

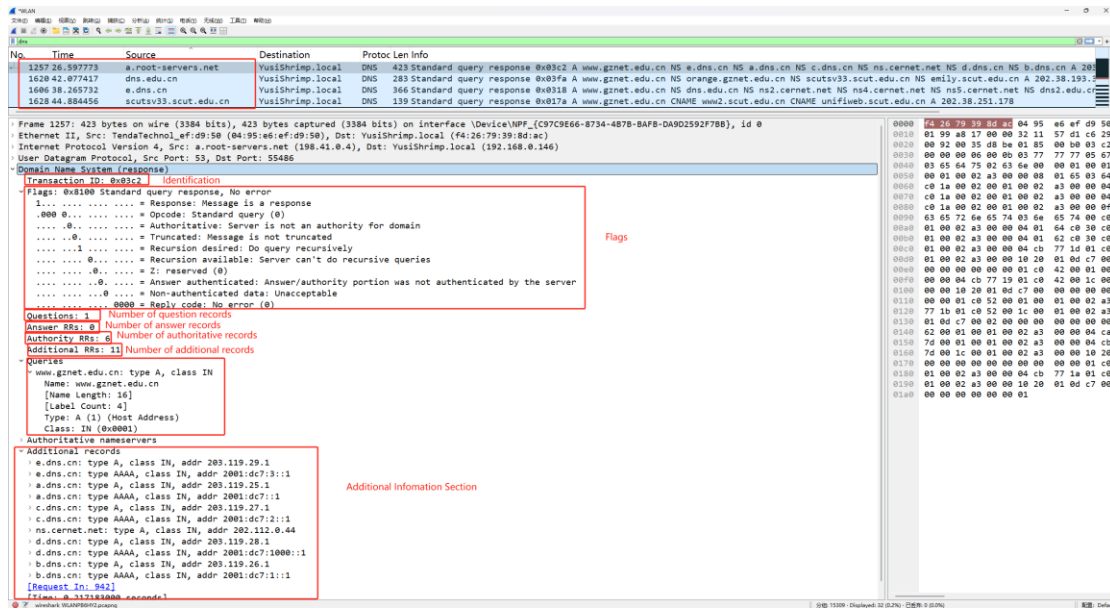


Figure 23: A DNS query packet

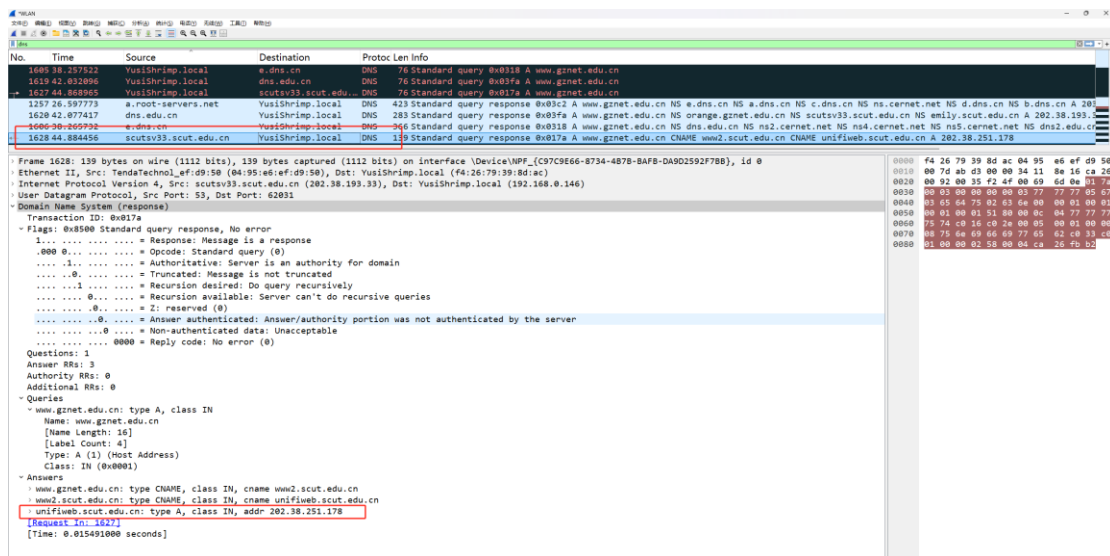


Figure 24: Final DNS response packet

According to the above operation, I can draw the figure that shows the sequence of remote nameservers I contacted and the domain for which they are responsible.

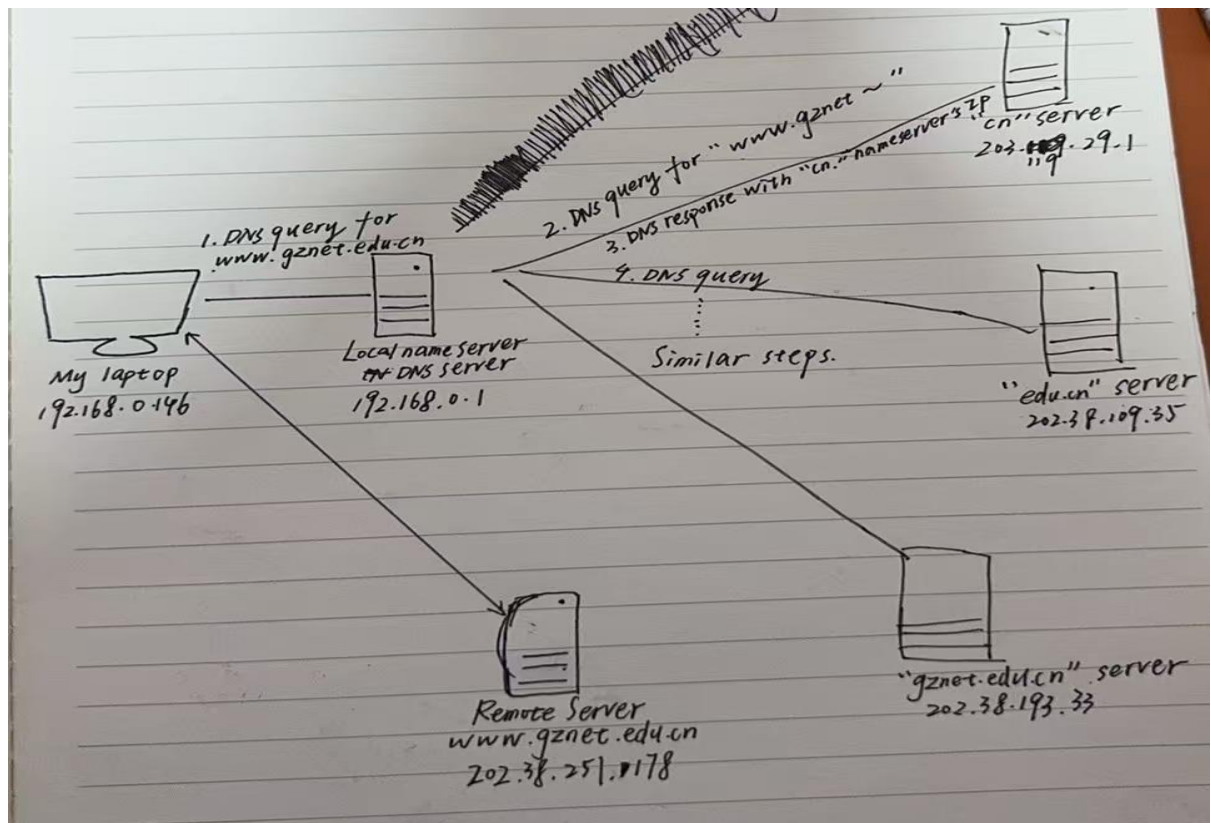


Figure 25: Contacted sequence of my host and nameservers

Finally, I also tried to trace DNS path automatically using dig +trace www.gznet.edu.cn, makes iterative queries to resolve the domain name.

```
C:\WINDOWS\system32\ x + v
C:\Users\86136>dig +trace www.genet.edu.cn
; <<>> Dig 9.4.3 <<>> +trace www.genet.edu.cn
;; global options: printcmd
. 124267 IN NS i.root-servers.net.
. 124267 IN NS a.root-servers.net.
. 124267 IN NS l.root-servers.net.
. 124267 IN NS b.root-servers.net.
. 124267 IN NS f.root-servers.net.
. 124267 IN NS d.root-servers.net.
. 124267 IN NS j.root-servers.net.
. 124267 IN NS e.root-servers.net.
. 124267 IN NS h.root-servers.net.
. 124267 IN NS m.root-servers.net.
. 124267 IN NS c.root-servers.net.
. 124267 IN NS g.root-servers.net.
. 124267 IN NS k.root-servers.net.
;; Received 504 bytes from 192.168.0.1#53(192.168.0.1) in 19 ms
cn. 172800 IN NS a.dns.cn.
cn. 172800 IN NS b.dns.cn.
cn. 172800 IN NS c.dns.cn.
cn. 172800 IN NS d.dns.cn.
cn. 172800 IN NS e.dns.cn.
cn. 172800 IN NS ns.cernet.net.
;; Received 381 bytes from 199.7.91.13#53(d.root-servers.net) in 221 ms
edu.cn. 172800 IN NS dns.edu.cn.
edu.cn. 172800 IN NS ns2.cernet.net.
edu.cn. 172800 IN NS ns4.cernet.net.
edu.cn. 172800 IN NS ns5.cernet.net.
edu.cn. 172800 IN NS dns2.edu.cn.
edu.cn. 172800 IN NS dns3.edu.cn.
;; Received 324 bytes from 203.119.25.1#53(a.dns.cn) in 42 ms
edu.cn. 21600 IN SOA dns.edu.cn. hostmaster.net.edu.cn. 2024120512 7200 1800 604800 21600
;; Received 181 bytes from 103.137.60.203#53(ns2.cernet.net) in 43 ms
C:\Users\86136>
```

The entire domain name resolution process is completed iteratively in one command

Figure 26: Using dig +trace command to resolve the name iteratively

3. Answer the questions

Q1: How many bits long is the Transaction ID? Based on this length, take your best guess as to how likely it is that concurrent transactions will use the same transaction ID.

A1: The Transaction ID in a DNS message is 16 bits long. Given that it's a 16-bit number, it can have 2^{16} possible values. The likelihood of concurrent transactions using the same Transaction ID depends on the number of requests made in a short time frame and the mechanism used by the DNS software to generate these IDs.

Q2: Which flag bit and what values signifies whether the DNS message is a query or response?

A2: The flag bit that signifies whether the DNS message is a query or a response is the first bit in Flags field in the DNS header. If the value is 0, it's a query; if it's 1, it's a response.

Q3: How many Bytes long is the entire DNS header?

A3: The entire DNS header is 12 Bytes long.

Q4: For the initial response, in what section are the names of the nameservers carried? What is the Type of the records that carry nameserver names?

A4: In the initial response, the names of the nameservers are carried in the "Authority" section of the DNS message. The Type of records that carry nameserver names is NS (Name Server).

Q5: Similarly, in what section are the IP addresses of the nameservers carried, and what is the Type of the records that carry the IP addresses?

A5: The IP addresses of the nameservers are carried in the "Additional" section of the DNS message, and the Type of records that carry the IP addresses is A for IPv4 and AAAA for IPv6.

Q6: For the final response, in what section is the IP address of the domain name carried?

A6: For the final response, the IP address of the domain name is carried in the "Answer" section of the DNS message.

Q7: What's the meaning of CNAME? And what happens if two CNAMEs points to each other?

A7: CNAME stands for Canonical Name. It is a type of resource record in the Domain Name System (DNS) that specifies that the domain name is an alias of another, canonical domain name. If two CNAMEs point to each other, it creates a loop, which is an erroneous configuration and can result in resolution failure as the query can go into an infinite loop.

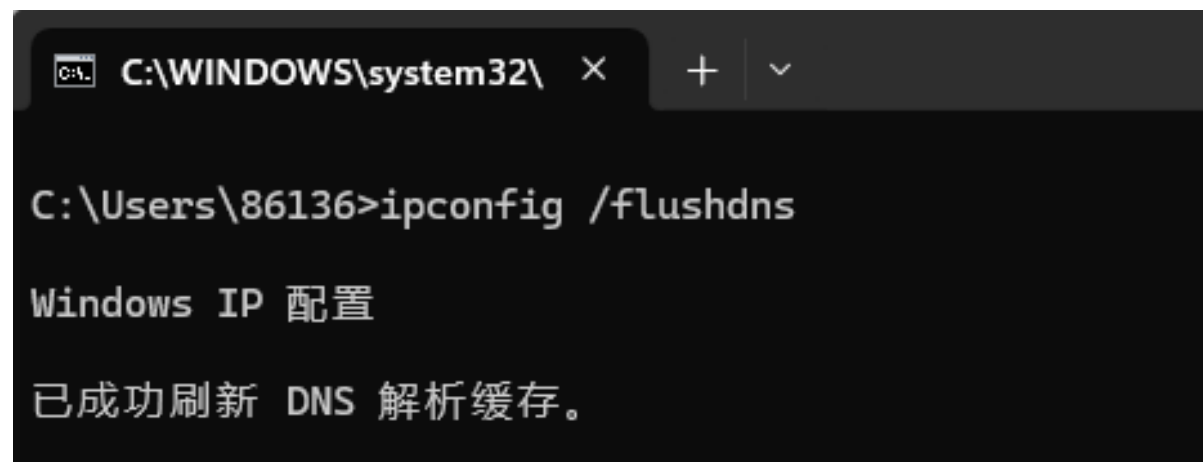
3) Task 3: DNS cache and Host file

1. Experiment steps:

- [1] **Step 1:** On Windows, execute “ipconfig /flushdns” command to clear DNS cache.
- [2] **Step 2:** Launch Wireshark to capture packets and setup filter condition to extract both DNS and ICMP protocol, e.g., “dns or icmp”.
- [3] **Step 3:** Execute “ping www.baidu.com” command, and then execute “ipconfig /displaydns” command to display DNS cache. Locate the corresponding records about the domain name in DNS cache.
- [4] **Step 4:** Execute “ping www.baidu.com” command again.
- [5] **Step 5:** Stop capturing, analyze the captured trace and the contents in DNS caches.
- [6] **Step 6:** According to your trace, draw a figure of the message interaction process (i.e., sequence of messages) in this task, including both ICMP and DNS query/reply messages.
- [7] **Step 7:** You can also check the usage “hosts” file using ping and Wireshark.
- [8] **Step 8:** Assign the correct IP address for a domain (e.g., www.baidu.com or www.jnu.edu.cn) in the host file, clear the DNS cache (i.e., Step 1), and then use ping and Wireshark to check whether the host file is working (e.g., whether the ping works and whether you can capture DNS request for the domain).
- [9] **Step 9:** Assign an incorrect IP address for a domain (e.g., www.baidu.com or www.jnu.edu.cn) in the host file, and then check whether you can ping the domain.

2. Results:

Execute sequentially “ipconfig /flushdns”, “ping www.baidu.com”, “ipconfig /displaydns” and “ping www. baidu.com” four commands.



```
C:\WINDOWS\system32\ x + v

C:\Users\86136>ipconfig /flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。
```

```
C:\Users\86136>ping www.baidu.com
```

```
正在 Ping www.a.shifen.com [183.240.98.198] 具有 32 字节的数据:  
来自 183.240.98.198 的回复: 字节=32 时间=13ms TTL=53  
来自 183.240.98.198 的回复: 字节=32 时间=13ms TTL=53  
来自 183.240.98.198 的回复: 字节=32 时间=24ms TTL=53  
来自 183.240.98.198 的回复: 字节=32 时间=34ms TTL=53
```

```
183.240.98.198 的 Ping 统计信息:
```

```
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),  
往返行程的估计时间(以毫秒为单位):  
    最短 = 13ms, 最长 = 34ms, 平均 = 21ms
```

```
C:\WINDOWS\system32\ x + v  
C:\Users\86136>ipconfig /displaydns
```

```
Windows IP 配置
```

```
www.baidu.com
```

```
-----  
记录名称 . . . . . : www.baidu.com  
记录类型 . . . . . : 5  
生存时间 . . . . . : 44  
数据长度 . . . . . : 8  
部分 . . . . . : 答案  
CNAME 记录 . . . . . : www.a.shifen.com
```

```
记录名称 . . . . . : www.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 答案  
A (主机)记录 . . . . . : 183.240.98.198
```

```
记录名称 . . . . . : www.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 答案  
A (主机)记录 . . . . . : 183.240.98.161
```

```
记录名称 . . . . . : ns3.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 其他  
A (主机)记录 . . . . . : 36.155.132.12
```

```
记录名称 . . . . . : ns3.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 其他  
A (主机)记录 . . . . . : 153.3.238.162
```

```
记录名称 . . . . . : ns4.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 其他  
A (主机)记录 . . . . . : 14.215.177.229
```

```
记录名称 . . . . . : ns4.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 其他  
A (主机)记录 . . . . . : 111.20.4.28
```

```
记录名称 . . . . . : ns5.a.shifen.com  
记录类型 . . . . . : 1  
生存时间 . . . . . : 44  
数据长度 . . . . . : 4  
部分 . . . . . : 其他  
A (主机)记录 . . . . . : 180.76.76.95
```

```
C:\Users\86136>ping www.baidu.com

正在 Ping www.a.shifen.com [183.240.98.198] 具有 32 字节的数据:
来自 183.240.98.198 的回复: 字节=32 时间=11ms TTL=53
来自 183.240.98.198 的回复: 字节=32 时间=12ms TTL=53
来自 183.240.98.198 的回复: 字节=32 时间=10ms TTL=53
来自 183.240.98.198 的回复: 字节=32 时间=13ms TTL=53

183.240.98.198 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 10ms, 最长 = 13ms, 平均 = 11ms
```

Figure 27: Execute four commands

During the execution of the script, I configured the Wireshark filter to capture packets related to the “www.baidu.com” site using the DNS and ICMP protocols. The results are as follows:



Figure 28: Check the Wireshark capture about cleaning DNS cache and fetching domain address operations

After executing the `ipconfig /flushdns` command, the DNS cache in this host is cleared. When the ping command is issued, the host needs to issue a DNS query to learn the IP address of `www.baidu.com`, then it can send ICMP protocol packets to the remote server in the ping command. When the ping command is executed for the second time, since the DNS cache has been stored in the host, there is no need to initiate a DNS query to the domain name server, so no data packets using the DNS protocol are captured.

The message transmission process that occurs during the execution of the entire script is as shown in the figure below:

```

第一次 Ping:
Your Computer --> Local DNS Server : DNS Query (www.baidu.com)
Local DNS Server --> Remote Server : DNS Query Forward
Remote Server --> Local DNS Server : DNS Response (IP Address)
Local DNS Server --> Your Computer : DNS Response (IP Address)
Your Computer --> www.baidu.com : ICMP Request
www.baidu.com --> Your Computer : ICMP Reply

第二次 Ping (使用缓存):
Your Computer --> www.baidu.com : ICMP Request
www.baidu.com --> Your Computer : ICMP Reply

```

Figure 29: Contacted sequence among my computer and nameservers

Next, I tried to modify the hosts file under the Windows system, adding www.baidu.com and its correct IP address to the hosts file and saving it:

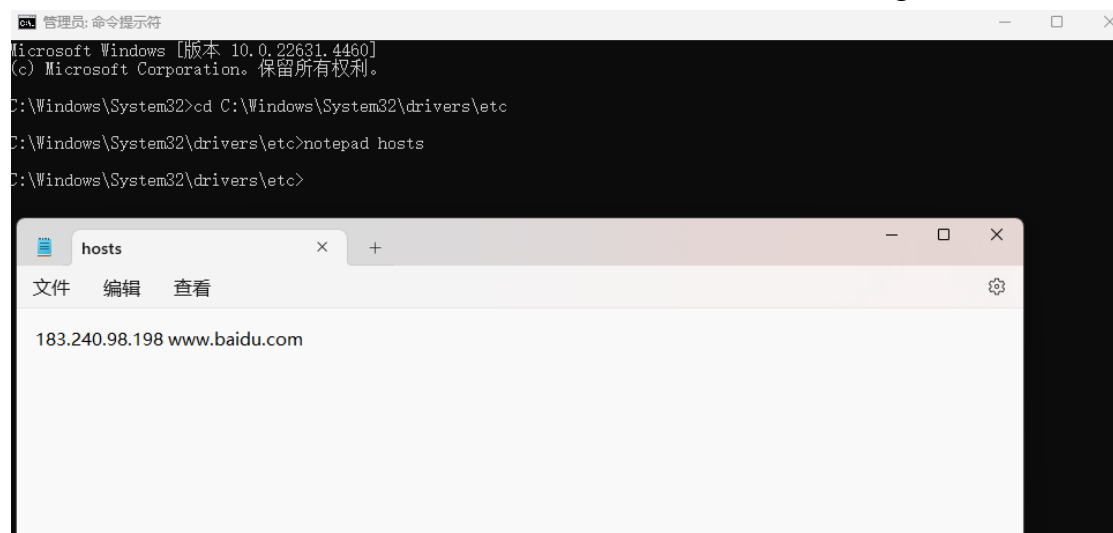


Figure 30: Add domain name to the hosts file with corresponding correct IP address

Then execute the same command as above.



```
www.baidu.com
-----
没有 AAAA 类型的记录

www.baidu.com
-----
记录名称 . . . . . : www.baidu.com
记录类型 . . . . . : 1
生存时间 . . . . . : 506204
数据长度 . . . . . : 4
部分 . . . . . : 答案
A (主机)记录 . . . . : 183.240.98.198
```

Figure 31: Execute the commands

After that, I modified the hosts file again, added www.baidu.com and loopback address (127.0.0.1) to the hosts file, commented out the correct IP address filled in before and finally save the file:

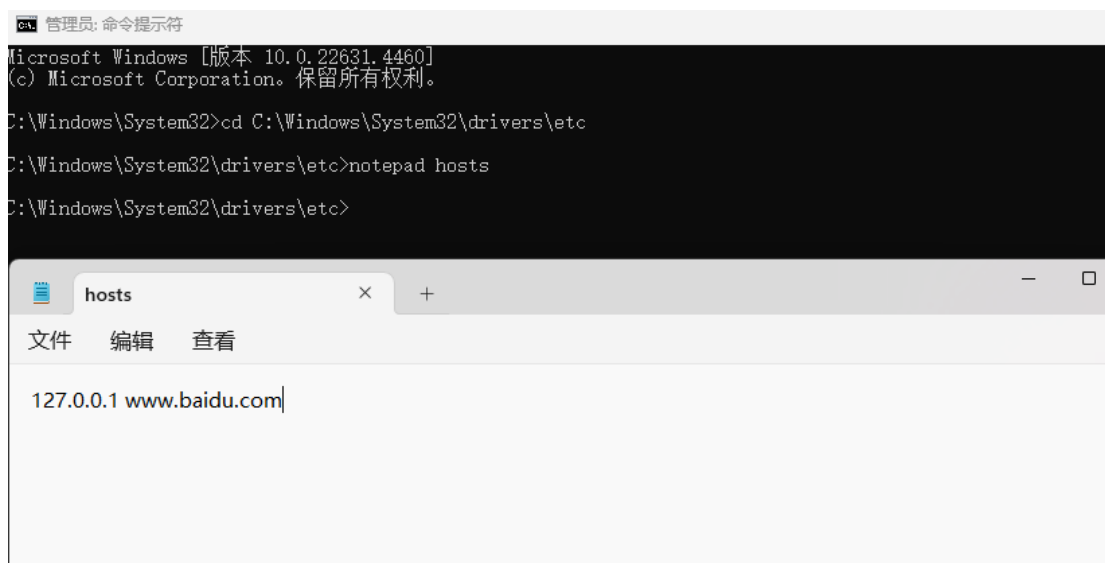


Figure 32: Change domain name to the hosts file

Then execute the same command as above again.


```
C:\WINDOWS\system32\ x + v

C:\Users\86136>ipconfig /flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。

C:\Users\86136>ping www.baidu.com

正在 Ping www.baidu.com [127.0.0.1] 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128

127.0.0.1 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms

C:\Users\86136>ipconfig /displaydns

最短 = 0ms, 最长 = 0ms, 平均 = 0ms

C:\Users\86136>ipconfig /displaydns

Windows IP 配置

www.baidu.com
-----
没有 AAAA 类型的记录

www.baidu.com
-----
记录名称. . . . . : www.baidu.com
记录类型. . . . . : 1
生存时间. . . . . : 505983
数据长度. . . . . : 4
部分. . . . . : 答案
A (主机)记录 . . . . : 127.0.0.1

1.0.0.127.in-addr.arpa
-----
记录名称. . . . . : 1.0.0.127.in-addr.arpa.
记录类型. . . . . : 12
生存时间. . . . . : 505983
数据长度. . . . . : 8
部分. . . . . : 答案
PTR 记录 . . . . . : www.baidu.com
```

Figure 33: Execute the commands again

In Wireshark, we can see that the captured data packets are only when www.baidu.com and its corresponding correct IP address are in the hosts file. When www .baidu.com and its loopback address are in the hosts file, I cannot capture

ICMP packets on the Ethernet interface. At the same time, no DNS request and reply packets are captured. This means that the priority of the hosts file is higher than DNS cache:

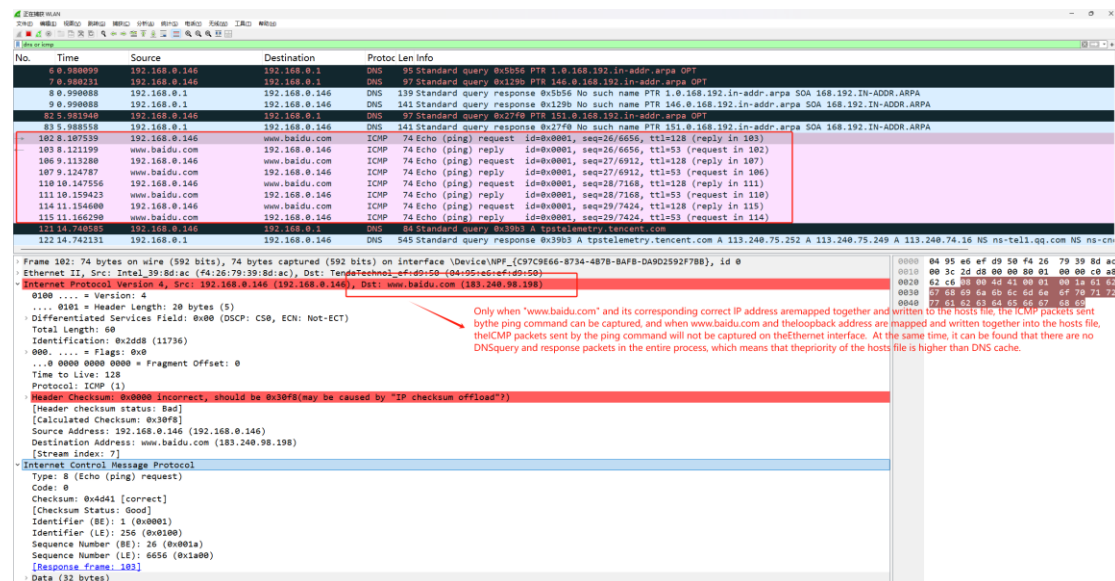


Figure 34: Check the Wireshark capture about hosts file modifying operations

3. Answer the questions

Q1: Will the second ping trigger DNS queries?

A1: The DNS cache already contains the record for “www.baidu.com” from a previous query, a second ping to the same address will not trigger another DNS query. The system will use the IP address from the cache.

Q2: Describe the usage of DNS cache.

A2: DNS cache serves to store the addresses of recently queried domain names. This reduces the need for repeated DNS server queries, thereby speeding up the resolution of frequently accessed domain names and reducing internet traffic.

Q3: What’s the priority of hosts file and DNS cache on your Operating System?

A3: On Windows, the “hosts” file has a higher priority than the DNS cache. When resolving domain names, Windows will first check the “hosts” file, and if an entry is found, it will use that information instead of querying the DNS cache or a DNS server.

4. Others

4.1 Personal thoughts

In this lab, I utilized the DNS protocol to capture various data packets, including both DNS query and response packets, while examining the fields within the DNS header. During the experiment, I performed domain name queries, manual resolution, and automatic iterative resolution using commands like "nslookup." Additionally, I explored the content and function of the DNS cache and gained an

understanding of the role the hosts file plays in domain name resolution. Overall, this lab enhanced and deepened my understanding of how the DNS protocol operates.