

# CS 305 Lab Tutorial

## Lab 5 DNS

Dept. Computer Science and Engineering  
Southern University of Science and Technology

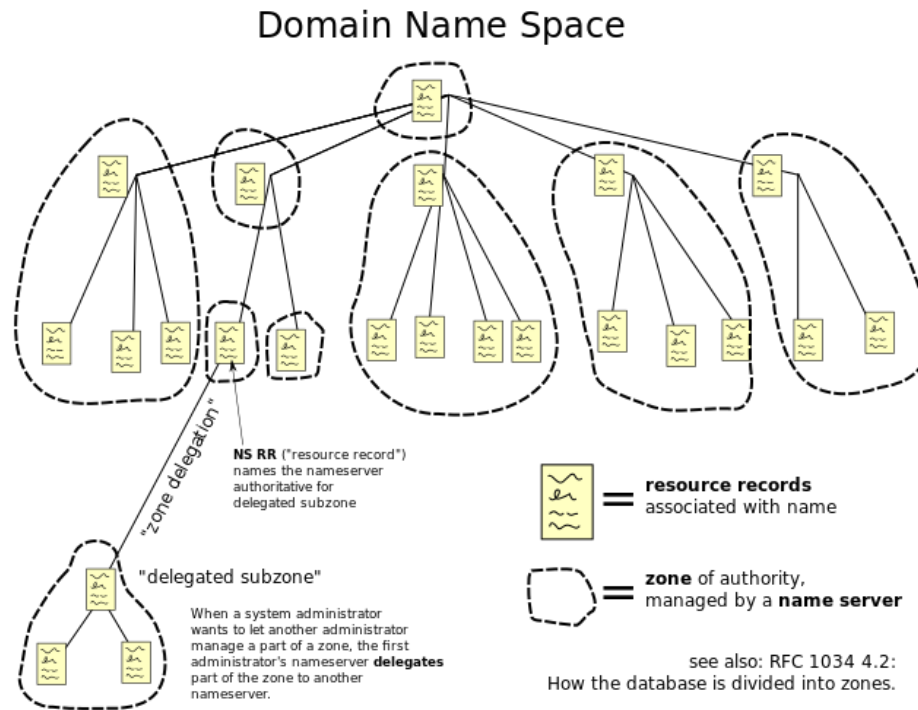
# Topic

- DNS
  - DNS Message Structure
  - DNS Message head
  - RR in DNS
- EDNS (aka. Extension mechanisms for DNS)
  - DNSSEC
- Tool : dig

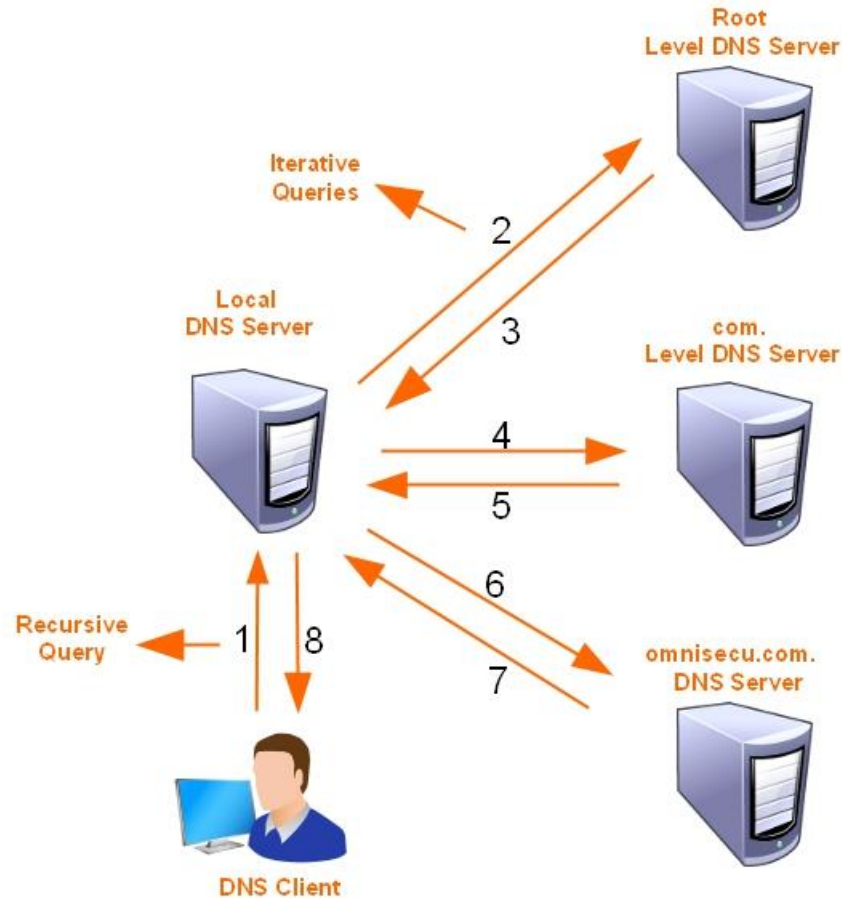
# Part A.1

## Domain Name System

- DNS is a distributed database.



# Recursive/Iterative Query



# Part A.2

## DNS Message Structure

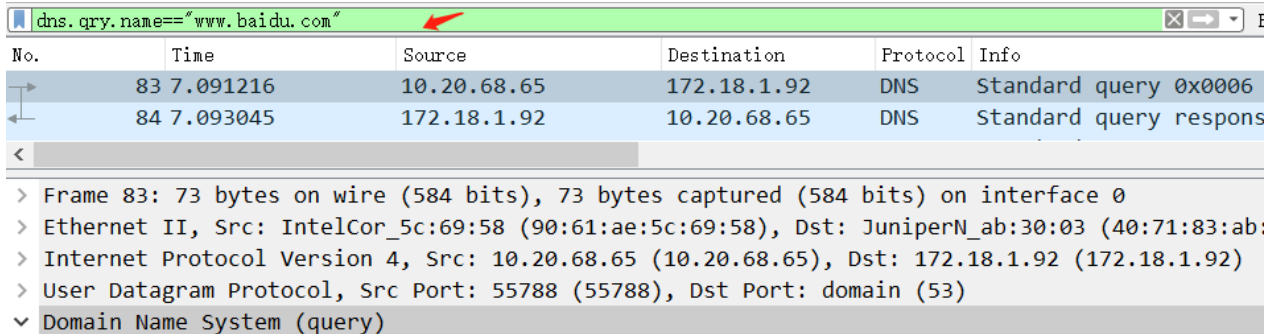
Header	
Question	the question for the name server
Answer	RRs answering the question
Authority	RRs pointing toward an authority
Additional	RRs holding additional information

0	15	16	31	
Transaction ID ( 会话标识 )		Flags ( 标志 )		} Header
Questions ( 问题数 )		Answer RRs ( 回答 资源记录数 )		
Authority RRs ( 授权 资源记录数 )		Additional RRs ( 附加 资源记录数 )		
Queries ( 查询问题区域 )				
Answers ( 回答区域 )				
Authoritative nameservers ( 授权区域 )				
Additional records ( 附加 区域 )				

DNS协议报文格式

# A query message of DNS

nslookup www.baidu.com



No.	Time	Source	Destination	Protocol	Info
83	7.091216	10.20.68.65	172.18.1.92	DNS	Standard query 0x0006
84	7.093045	172.18.1.92	10.20.68.65	DNS	Standard query response

> Frame 83: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface 0  
> Ethernet II, Src: IntelCor\_5c:69:58 (90:61:ae:5c:69:58), Dst: JuniperN\_ab:30:03 (40:71:83:ab:30:03)  
> Internet Protocol Version 4, Src: 10.20.68.65 (10.20.68.65), Dst: 172.18.1.92 (172.18.1.92)  
> User Datagram Protocol, Src Port: 55788 (55788), Dst Port: domain (53)  
✓ Domain Name System (query)

Transaction ID: 0x0006

✓ Flags: 0x0100 Standard query

- 0... .. = Response: Message is a query
- .000 0... .. = Opcode: Standard query (0)
- .... ..0. .... = Truncated: Message is not truncated
- .... ...1 .... = Recursion desired: Do query recursively
- .... .... .0.. .... = Z: reserved (0)
- .... .... ...0 .... = Non-authenticated data: Unacceptable

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

just 1 question with no answer

✓ Queries

- ✓ www.baidu.com: type A, class IN
  - Name: www.baidu.com
  - [Name Length: 13]
  - [Label Count: 3]
  - Type: A (Host Address) (1)
  - Class: IN (0x0001)

[Response In: 84]

# A response message of DNS

nslookup www.baidu.com

dns.qry.name=="www.baidu.com"

No.	Time	Source	Destination	Protocol	Info
84	7.093045	172.18.1.92	10.20.68.65	DNS	Standard query response 0x0006

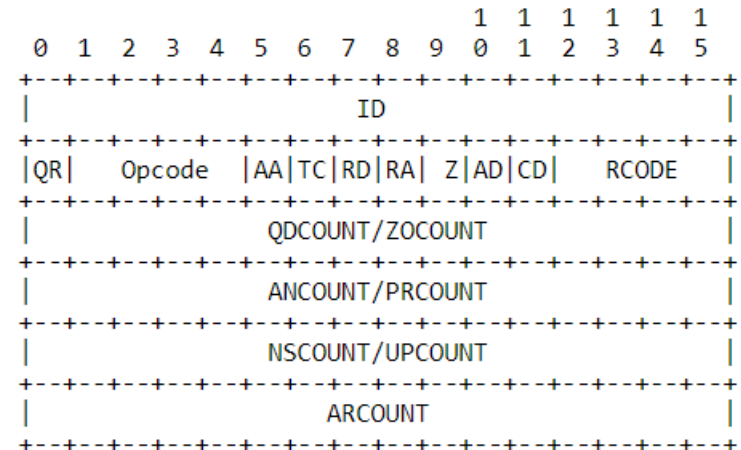
> Frame 84: 286 bytes on wire (2288 bits), 286 bytes captured (2288 bits) on interface 0  
> Ethernet II, Src: JuniperN\_ab:30:03 (40:71:83:ab:30:03), Dst: IntelCor\_5c:69:58 (90:61:ae:5c:69:58)  
> Internet Protocol Version 4, Src: 172.18.1.92 (172.18.1.92), Dst: 10.20.68.65 (10.20.68.65)  
> User Datagram Protocol, Src Port: domain (53), Dst Port: 55788 (55788)  
✓ Domain Name System (response)  
Transaction ID: 0x0006  
✓ Flags: 0x8180 Standard query response, No error  
① ... .. = Response: Message is a response  
... .. = Opcode: Standard query (0)  
... .. = Authoritative: Server is not an authority for domain  
... .. = Truncated: Message is not truncated  
... ..1 ... = Recursion desired: Do query recursively  
... .. 1... = Recursion available: Server can do recursive queries  
... .. .0.. = Z: reserved (0)  
... .. ..0. = Answer authenticated: Answer/authority portion was not authenticated by the :  
... .. ...0 = Non-authenticated data: Unacceptable  
... .. ... 0000 = Reply code: No error (0)  
Questions: 1  
Answer RRs: 3  
Authority RRs: 5  
Additional RRs: 4  
> Queries  
> Answers  
> Authoritative nameservers  
> Additional records  
[\[Request In: 83\]](#)  
[Time: 0.001829000 seconds]

# RFC 2929 DNS Message Headers

## Domain Name System (DNS) IANA Considerations

- Set QR bit to 0 indicates the header is a query, otherwise is a response.
- OpCode 0 indicates this is a standard query.
- AA, TC, RD, RA, AD, CD stands for Authoritative Answer, Truncated, Recursion Desired, Recursion Available, Checking Disabled.
- Z is a reserved flag.

OpCode	Name	Reference
0	Query	<a href="#">[RFC 1035]</a>
1	IQuery (Inverse Query)	<a href="#">[RFC 1035]</a>
2	Status	<a href="#">[RFC 1035]</a>
3	available for assignment	
4	Notify	<a href="#">[RFC 1996]</a>
5	Update	<a href="#">[RFC 2136]</a>
6-15	available for assignment	





# Example Structure Code in C:

```
struct DNS_HEADER {    //DNS header structure
    unsigned short id;    // identification number

    unsigned char qr :1;    // query/response flag
    unsigned char opcode :4; // purpose of message
    unsigned char aa :1;    // authoritative answer
    unsigned char tc :1;    // truncated message
    unsigned char rd :1;    // recursion desired
    unsigned char ra :1;    // recursion available
    unsigned char z :1;    // its z! reserved
    unsigned char ad :1;    // authenticated data
    unsigned char cd :1;    // checking disabled
    unsigned char rcode :4; // response code

    unsigned short q_count; // number of question entries
    unsigned short ans_count; // number of answer entries
    unsigned short auth_count; // number of authority entries
    unsigned short add_count; // number of resource entries
};
```

# Decode Message Header in Python

```
class DNSHeader:
```

```
    Struct = struct.Struct('!6H')
```

```
    def __init__(self):
```

```
        self.__dict__ = {
```

```
            field: None
```

```
            for field in ('ID', 'QR', 'OpCode', 'AA', 'TC', 'RD', 'RA', 'Z',  
                          'RCode', 'QDCount', 'ANCount', 'NSCount', 'ARCount')}]
```

```
    def parse_header(self, data):
```

```
        self.ID, misc, self.QDCount, self.ANcount, self.NScount, self.NScount = DNSHeader.Struct.unpack_from(data)
```

```
        self.QR = (misc & 0x8000) != 0
```

```
        self.OpCode = (misc & 0x7800) >> 11
```

```
        self.AA = (misc & 0x0400) != 0
```

```
        self.TC = (misc & 0x200) != 0
```

```
        self.RD = (misc & 0x100) != 0
```

```
        self.RA = (misc & 0x80) != 0
```

```
        self.Z = (misc & 0x70) >> 4 # Never used
```

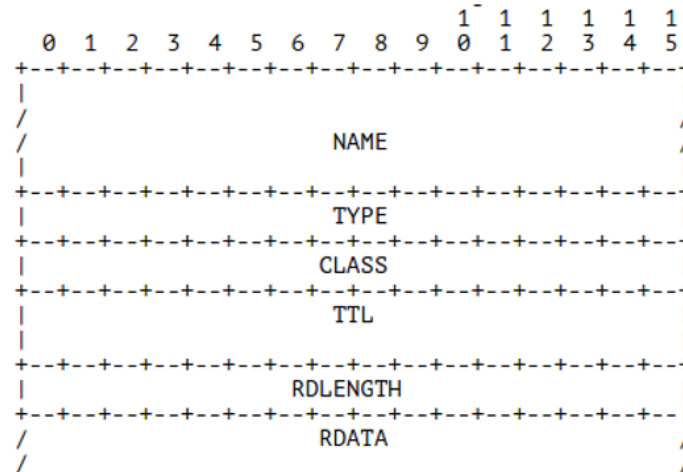
```
        self.RCode = misc & 0xF
```

```
    def __str__(self):
```

```
        return '<DNSHeader {}>'.format(str(self.__dict__))
```

# Part A.3

## RR in DNS



Resource record (RR) fields

Field	Description	Length (octets)
NAME	Name of the node to which this record pertains	Variable
TYPE	Type of RR in numeric form (e.g., 15 for MX RRs)	2
CLASS	Class code	2
TTL	Count of seconds that the RR stays valid (The maximum is $2^{31}-1$ , which is about 68 years)	4
RDLENGTH	Length of RDATA field (specified in octets)	2
RDATA	Additional RR-specific data	Variable, as per RDLENGTH

# RRs of Answers

nslookup www.baidu.com

No.	Time	Source	Destination	Protocol	Info
84	7.093045	172.18.1.92	10.20.68.65	DNS	Standard query response 0x0006
< >					
Domain Name System (response)					
Transaction ID: 0x0006					
> Flags: 0x8180 Standard query response, No error					
Questions: 1					
Answer RRs: 3					
Authority RRs: 5					
Additional RRs: 4					
> Queries					
Answers					
www.baidu.com: type CNAME, class IN, cname www.a.shifen.com					
Name: www.baidu.com					
Type: CNAME (Canonical NAME for an alias) (5)					
Class: IN (0x0001)					
Time to live: 77					
Data length: 15					
CNAME: www.a.shifen.com					
www.a.shifen.com: type A, class IN, addr 14.215.177.38					
Name: www.a.shifen.com					
Type: A (Host Address) (1)					
Class: IN (0x0001)					
Time to live: 168					
Data length: 4					
Address: www.a.shifen.com (14.215.177.38)					
www.a.shifen.com: type A, class IN, addr 14.215.177.39					
Name: www.a.shifen.com					
Type: A (Host Address) (1)					
Class: IN (0x0001)					
Time to live: 168					
Data length: 4					
Address: www.a.shifen.com (14.215.177.39)					
> Authoritative nameservers					

all the answers share  
the same structure:  
name,type,class,ttl  
and length

# RRs of authoritative name servers

nslookup www.baidu.com

```
✓ Domain Name System (response)
  Transaction ID: 0x0006
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 3
  Authority RRs: 5
  Additional RRs: 4
  > Queries
  > Answers
  ✓ Authoritative nameservers
    ✓ a.shifen.com: type NS, class IN, ns ns3.a.shifen.com
      Name: a.shifen.com
      Type: NS (authoritative Name Server) (2)
      Class: IN (0x0001)
      Time to live: 66
      Data length: 6
      Name Server: ns3.a.shifen.com
    > a.shifen.com: type NS, class IN, ns ns2.a.shifen.com
    > a.shifen.com: type NS, class IN, ns ns1.a.shifen.com
    > a.shifen.com: type NS, class IN, ns ns5.a.shifen.com
    > a.shifen.com: type NS, class IN, ns ns4.a.shifen.com
  > Additional records
  [Request In: 83]
  [Time: 0.001829000 seconds]
```

the value of rdata depend on  
the type

# RRs of Additional records

nslookup www.baidu.com

```
dns.gry.name=="www.baidu.com" Expression...
No.      Time      Source      Destination  Protocol Info
84 7.093045 172.18.1.92 10.20.68.65  DNS      Standard query response 0x0006

> Ethernet II, Src: JuniperN_ab:30:03 (40:71:83:ab:30:03), Dst: IntelCor_5c:69:58 (90:61:ae:5c:69:58)
> Internet Protocol Version 4, Src: 172.18.1.92 (172.18.1.92), Dst: 10.20.68.65 (10.20.68.65)
> User Datagram Protocol, Src Port: domain (53), Dst Port: 55788 (55788)
< Domain Name System (response)
  Transaction ID: 0x0006
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 3
  Authority RRs: 5
  Additional RRs: 4
  > Queries
  > Answers
  > Authoritative nameservers
  < Additional records
    < ns1.a.shifen.com: type A, class IN, addr 61.135.165.224
      Name: ns1.a.shifen.com
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 423
      Data length: 4
      Address: ns1.a.shifen.com (61.135.165.224)
    < ns2.a.shifen.com: type A, class IN, addr 220.181.33.32
      Name: ns2.a.shifen.com
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 4
      Data length: 4
      Address: ns2.a.shifen.com (220.181.33.32)
    > ns3.a.shifen.com: type A, class IN, addr 112.80.255.253
    > ns4.a.shifen.com: type A, class IN, addr 14.215.177.229
[Request in: 83]
[Time: 0.001829000 seconds]
```

# Part B

## EDNS (aka. Extension mechanisms for DNS)

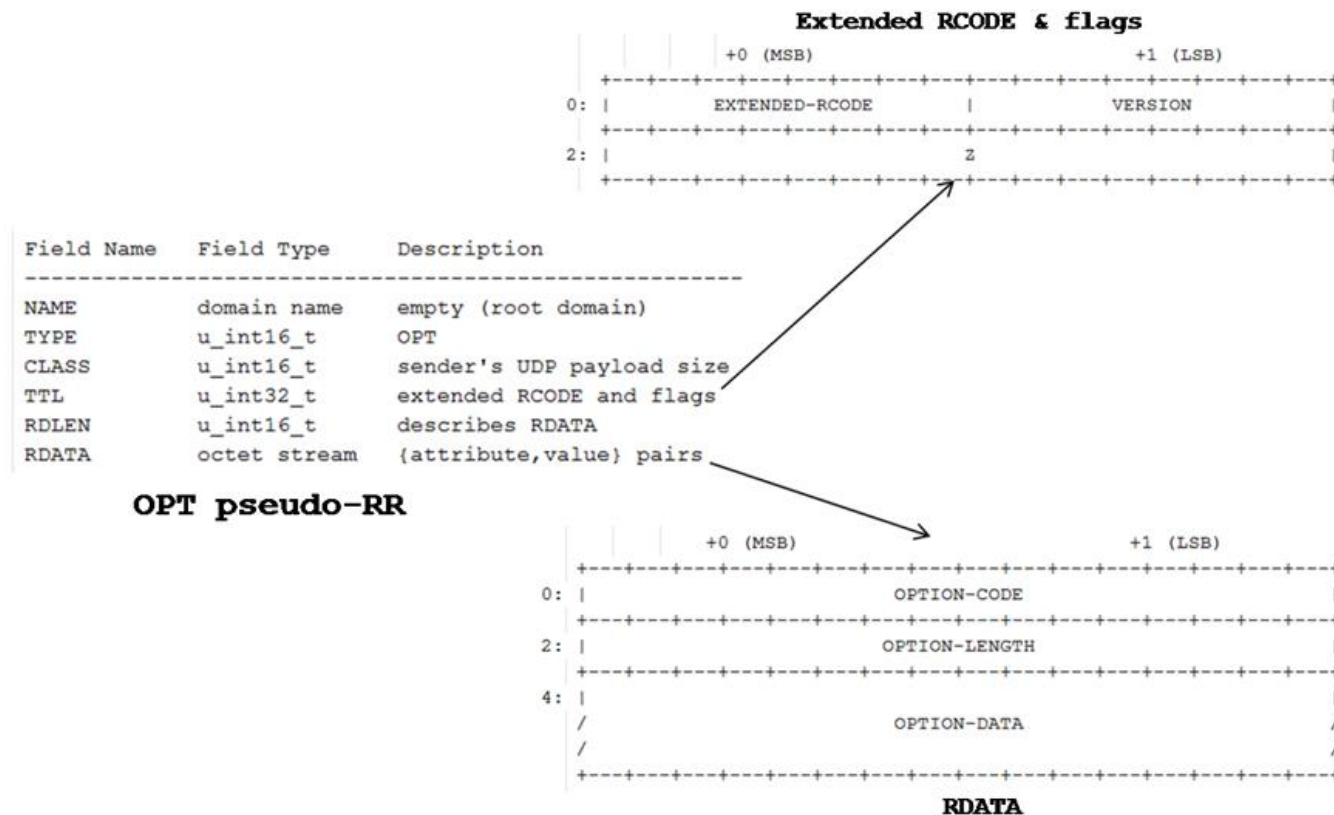
EDNS: a backward compatible mechanisms for allowing the DNS protocol to grow.

- The Domain Name System's wire protocol includes a number of fixed fields whose range has been or soon will be exhausted and does not allow clients to advertise their capabilities to servers
- DNS (see [RFC1035]) specifies a Message Format and within such messages there are standard formats for encoding options, errors, and name compression. The maximum allowable size of a DNS Message is fixed.
- Many of DNS's protocol limits are too small for uses which are or which are desired to become common. There is no way for implementations to advertise their capabilities.

<https://tools.ietf.org/html/rfc2671>

# EDNS

One **OPT pseudo-RR** can be added to the **additional data section** of either a request or a response. An OPT is called a pseudo-RR because it pertains to a particular transport level message and not to any actual DNS data.





# EDNS query

## Domain Name System (query)

Transaction ID: 0xe9d8

> Flags: 0x0120 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 1

> Queries

### Additional records

#### <Root>: type OPT

Name: <Root>

Type: OPT (41)

UDP payload size: 4096

Higher bits in extended RCODE: 0x00

EDNS0 version: 0

#### Z: 0x0000

0... .... = DO bit: Cannot handle DNSSEC security RRs

.000 0000 0000 0000 = Reserved: 0x0000

Data length: 12

> Option: COOKIE

Field Name	Field Type	Description
NAME	domain name	empty (root domain)
TYPE	u_int16_t	OPT
CLASS	u_int16_t	sender's UDP payload size
TTL	u_int32_t	extended RCODE and flags
RDLEN	u_int16_t	describes RDATA
RDATA	octet stream	{attribute,value} pairs

# EDNS response

## Domain Name System (response)

Transaction ID: 0xe9d8

> Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 3

Authority RRs: 5

Additional RRs: 5

> Queries

> Answers

> Authoritative nameservers

### Additional records

> ns1.a.shifen.com: type A, class IN, addr 61.135.165.224

> ns2.a.shifen.com: type A, class IN, addr 220.181.33.32

> ns3.a.shifen.com: type A, class IN, addr 112.80.255.253

> ns5.a.shifen.com: type A, class IN, addr 180.76.76.95

### <Root>: type OPT

Name: <Root>

Type: OPT (41)

UDP payload size: 4096

Higher bits in extended RCODE: 0x00

EDNS0 version: 0

### Z: 0x0000

0... .. = DO bit: Cannot handle DNSSEC security RRs

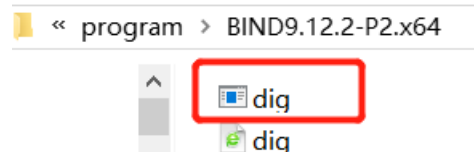
.000 0000 0000 0000 = Reserved: 0x0000

Data length: 0

Field Name	Field Type	Description
NAME	domain name	empty (root domain)
TYPE	u_int16_t	OPT
CLASS	u_int16_t	sender's UDP payload size
TTL	u_int32_t	extended RCODE and flags
RDLEN	u_int16_t	describes RDATA
RDATA	octet stream	{attribute,value} pairs

# dig(1)

- **dig** is a flexible tool for interrogating DNS name servers.
  - It performs DNS lookups and displays the answers that are returned from the name server(s) that were queried.
  - Most DNS administrators use **dig** to troubleshoot DNS problems because of its flexibility, ease of use and clarity of output.



Bind is a Toolset which includes dig as a component  
Bind could be get from <https://www.isc.org/bind/>

# Using dig(2)

A typical invocation of dig looks like: **dig @server name type**

**server** is **the name or IP address of the name server to query**. This can be an IPv4 address in dotted-decimal notation or an IPv6 address in colon-delimited notation. When the supplied server argument is a hostname, dig resolves that name before querying that name server.

**name** is **the name of the resource record that is to be looked up**.

## **type**

indicates **what type of query is required** — **ANY, A, MX, SIG**, etc. type can be any valid query type. If no type argument is supplied, dig will perform a lookup for an A record.

```
D:\>dig @ns2.sustech.edu.cn www.baidu.com a +short
www.a.shifen.com.
163.177.151.109
163.177.151.110

D:\>dig @ns2.sustech.edu.cn www.baidu.com cname +short
www.a.shifen.com.

D:\>dig @ns2.sustech.edu.cn www.baidu.com mx +short
www.a.shifen.com.
```

lots of useful options like: +tcp, +noedns, +bufsize, +trace, etc. have a try!

# lab assignment 5.1

- Make two DNS queries for “[www.baidu.com](http://www.baidu.com)”, with and without 'EDNS0' option respectively.
  - Screenshot on this command and its output
- capture the packages using Wireshark
  - what is the content of these query message
    - what's the destination IP address and destination port of these query?
    - Find the name, type and class of this query
    - what's the opcode of these query, what does it mean?
    - what's the difference of these two queries?
  - what is the content of the response messages
    - how to judge the response is legal or not?
    - Is there any answers, what's life time of each answer?
    - Is there any authority RRs, what's the type of each RR?

# lab assignment 5.2

- Make the query of “[www.sina.com.cn](http://www.sina.com.cn)” by using dig with option “+trace”
- Screenshot on the command and its output . answer the questions by analysis the packets:
  - is there any query with 'rd' field is set 0 ? Is there any relationship with “+trace” of dig?
  - how many root DNS servers, what are their name?
  - which server sent the the last response( which server answered the 'A' type value of [www.sina.com.cn](http://www.sina.com.cn)) ?
    - List its name, IP address and port.
    - What's the value of 'aa' field in this response message?
  - try the same query again
    - This time, is the last response from the same server as the last response in the previous query? If they are different, what is the reason for this? will it bring any benefits?

# lab 5.3 implement a 'local' DNS server

- Function:
  - Listen and accept DNS queries.
    - Support common query types: A, AAAA, CNAME, NS, MX
    - EDNS implementation is not required.
    - if 'RD' of query is set, implement recursive query.
    - if 'RD' is not set, forward query to a public DNS server.
  - Check out the response and send response to your clients.
  - Maintain a cache of DNS query-response of all results.
- Test method:
  - using dig sending query to this 'local' DNS server
- \*comments is MUST

# Tips: Using dns.resolver of python(1)

## Using pip to install dnspython

- pip is the package installer for Python. You can use pip to install packages from the Python Package Index and other indexes.

```
C:\Users\wzr>pip install dnspython
Collecting dnspython
  Downloading https://files.pythonhosted.org/packages/a6/72/209e18bdfedfd78c6994e9ec96981624a5ad7738524dd474237268422ct
/dnspython-1.15.0-py2.py3-none-any.whl (177kB)
    100% |#####| 184kB 18kB/s
Installing collected packages: dnspython
Successfully installed dnspython-1.15.0
```

## A demo of using query of dns.resolver

If 'pip' is not installed on your computer, get it from  
<https://pypi.org/project/pip/>

Get more infor about dnspython, get it from  
<https://pypi.org/project/dnspython/>

```
>>> import dns.resolver
>>> dns.resolver.query("www.baidu.com", 'a')
<dns.resolver.Answer object at 0x000002316AF22860>
>>> a = dns.resolver.query("www.baidu.com", 'a')
>>> a
<dns.resolver.Answer object at 0x000002316AF277F0>
>>> for i in a.response.answer:
...     for j in i.items:
...         print(j)
...
www.a.shifen.com.
163.177.151.110
163.177.151.109
>>>
```



# Tips: Using dns.resolver of python(2)

## query in dns.resolver of python

- `query(self, qname, rdtype=1, rdclass=1, tcp=False, source=None, raise_on_no_answer=True, source_port=0)`
  - Query nameservers to find the answer to the question.
  - The `qname`, `rdtype`, and `rdclass` parameters may be objects of the appropriate type, or strings that can be converted into objects of the appropriate type. E.g. For `rdtype` the integer 2 and the string 'NS' both mean to query for records with DNS rdata type NS.
- Parameters:
  - `qname` (`dns.name.Name` object or string) - the query name
  - `rdtype` (int or string) - the query type
  - `rdclass` (int or string) - the query class
  - `tcp` (bool) - use TCP to make the query (default is False).
  - `source` (IP address in dotted quad notation) - bind to this IP address (defaults to machine default IP).
  - `raise_on_no_answer` (bool) - raise `NoAnswer` if there's no answer (defaults is True).
  - `source_port` (int) - The port from which to send the message. The default is 0.

# Tips: UDP socket programming

```
udp_c.py x udp_s.py x
1 from socket import *
2 serverPort = 12000
3 serverSocket = socket(AF_INET, SOCK_DGRAM)
4 serverSocket.bind(('', serverPort))
5 print ("The server is ready to receive")
6 while True:
7     message, clientAddress = serverSocket.recvfrom(2048)
8     modifiedMessage = message.decode().upper()
9     serverSocket.sendto(modifiedMessage.encode(), clientAddress)
```

```
udp_c.py x udp_s.py x
1 from socket import *
2 serverName = '127.0.0.1'
3 serverPort = 12000
4 clientSocket = socket(AF_INET, SOCK_DGRAM)
5 message = input('Input lowercase sentence:')
6 clientSocket.sendto(message.encode(), (serverName, serverPort))
7 modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
8 print(modifiedMessage.decode())
9 clientSocket.close()
```

```
d:\python_test>python udp_s.py
The server is ready to receive
```

```
d:\python_test>python udp_c.py
Input lowercase sentence:azs
AZS
```