DNS_Relay

吴毅龙 PB19111749

实验思路

在`localhost`的53端口创建一个`socket`，接收和发送DNS请求或者响应都通过这个端口。

在该端口上接收到数据时，对其进行解析：

- 如果是一个DNS请求报文，有以下两种情况
  - 请求的域名不在配置文件中，则将DNS请求报文转发给一个可靠的DNS服务器，这里选择的是`114.114.114.114`DNS服务器。
  - 请求的域名在配置文件中，则生成DNS响应报文并发送给请求方。

    > If the queried name is in the list and its associated IP address is “0.0.0.0”, responds 0.0.0.0 to the client.

    > If the queried name is in the list and has a meaningful IP address associated, responds that IP address.

- 如果是一个DNS响应报文，则根据其transaction id转发到对应的地址

实现细节

本次实验顺利进行的基础是熟练掌握DNS请求和响应报文的格式，所有实现步骤包括报文的分类、报文数据的解析、响应报文的生成都是基于DNS报文格式进行的。

- `DNS_Relay_Server`类通过配置文件和外部地址来初始化。`load_file`方法通过读取配置文件的信息阿里装填`url_ip`字典，建立`url`与`ip`地址之间的映射关系

  ```
  def load_file(self,):
          f = open(self.cache_file,'r',encoding='utf-8')
          for line in f:
              ip,name = line.split(' ')
              self.url_ip[name.strip('\n')] = ip
          f.close()
  ```

  `run`方法则在`localhost`的53端口创建一个`socket`，接收和发送DNS请求或者响应都通过这个端口。

  ```
  def run(self):
          buffer_size = 512
          server_socket = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
          server_socket.bind(('',53))
          server_socket.setblocking(False)
          while True:
              try:
                  data,addr = server_socket.recvfrom(buffer_size)
                  threading.Thread(target=self.handle,args=
  (server_socket,data,addr)).start()
              except:
                  continue
  ```

`handle`则利用DNS报文头中的`QR`字段判断报文类型（请求报文&响应报文），具体判断方法则是根据DNS报文中`QR`字段的含义。做出相应的判断后的动作如前所述。

```
def handle(self,server_socket,data,addr):
        RecvDp = DNS_Packege(data)
        if RecvDp.QR == 0:           #是请求报文:
            if RecvDp.name not in self.url_ip:
                self.trans[RecvDp.ID] = addr
                server_socket.sendto(data, ('114.114.114.114', 53))
            else:
                ip = self.url_ip[RecvDp.name]
                data_to_send = RecvDp.generate_response(ip, ip == '0.0.0.0')
                server_socket.sendto(data_to_send, addr)
        if RecvDp.QR == 1:           #是响应报文:
            addr = self.trans[RecvDp.ID]
            server_socket.sendto(data, addr)
```

- `DNS_Packege`则用于解析和生成DNS报文。根据DNS报文的格式，对每一部分的字段进行数据解析，这里就需要使用按位操作来获取每一个字段的数据，包括`ID`、`flag`、资源记录数量。

```
Msg_arr = bytearray(data)
#ID
self.ID = (Msg_arr[0] << 8 ) + Msg_arr[1]
# FLAGS
self.QR = Msg_arr[2] >> 7
self.Opcode = (Msg_arr[2] & 0b01111000) >> 3
self.AA = (Msg_arr[2] & 0b00000100) >> 2
self.TC = (Msg_arr[2] & 0b00000010) >> 1
self.RD = Msg_arr[2] & 0b00000001
self.RA = (Msg_arr[3] & 0b10000000) >> 7
self.Z = (Msg_arr[3] & 0b01110000) >> 4
self.RCODE = Msg_arr[3] & 0b00001111
# 资源记录数量
self.QDCOUNT = (Msg_arr[4] << 8) + Msg_arr[5]
self.ANCOUNT = (Msg_arr[6] << 8) + Msg_arr[7]
self.NSCOUNT = (Msg_arr[8] << 8) + Msg_arr[9]
self.ARCOUNT = (Msg_arr[10] << 8) + Msg_arr[11]
```

对于`query`内容解析则更加需要熟悉，理解每一个label序列的含义并正确使用。每一个字符串前都是一个表示字符串长度的label，整个序列以全零串结束。

> 域名被编码为一些labels序列，每个labels包含一个字节表示后续字符串长度，以及这个字符串，以0长度和空字符串来表示域名结束。注意这个字段可能为奇数字节，不需要进行边界填充对齐。

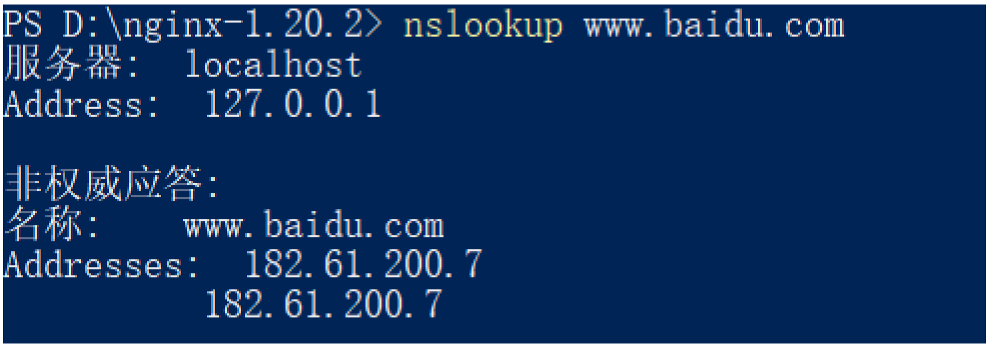
```
#query内容解析
i = 12
name = ''
length = int(Msg_arr[12])
while True:
    for j in range(i + 1, i + length + 1):
        name = name + chr(Msg_arr[j])
        i = i + length + 1
        length = int(Msg_arr[i])
        if length == 0:
            break
        else:
            name = name + '.'

self.name = name
self.name_length = len(name) + 2
```

`generate`方法则用于生成响应DNS报文，根据响应报文各个字段的含义为相应的字段赋值，最后将头部的各个域和问题域、回答域拼接在一起

```
def generate_response(self,ip,Intercepted):
        self.QR = 1    #响应时QR为1
        self.AA = 0    #该字段在响应报文中有效。值为 0 时，表示不是权威服务器。
        self.RA = 1    #该字段只出现在响应报文中。当值为 1 时，表示服务器支持递归查询。
        self.Z = 0     #保留字段，在所有的请求和应答报文中，它的值必须为 0。
        self.ANCOUNT = 1
        self.NSCOUNT = 0
        self.ARCOUNT = 0
        if not Intercepted:
            self.RCODE = 0
            res = bytearray(32 + self.name_length)
            res[0] = self.ID >> 8
            res[1] = self.ID % 256
            res[2] = (self.QR << 7) + (self.Opcode << 3) + (self.AA << 2) + (self.TC << 1) + self.RD
            res[3] = (self.RA << 7) + (self.Z << 4) + self.RCODE
            res[4] = self.QDCOUNT >> 8
            res[5] = self.QDCOUNT % 256
            res[6] = self.ANCOUNT >> 8
            res[7] = self.ANCOUNT % 8
            res[8] = self.NSCOUNT >> 8
            res[9] = self.NSCOUNT % 8
            res[10] = self.ARCOUNT >> 8
            res[11] = self.ARCOUNT % 8
            for i in range(12, 16 + self.name_length):
                res[i] = self.data[i]
            res[16 + self.name_length] = 0xc0
            res[17 + self.name_length] = 0x0c
            res[18 + self.name_length] = 0x00
            res[19 + self.name_length] = 0x01  #Type = A
            res[20 + self.name_length] = 0x00
            res[21 + self.name_length] = 0x01  #Class = IN
            for i in range(22 + self.name_length, 26 + self.name_length):
                res[i] = 0
            res[26 + self.name_length] = 0
            res[27 + self.name_length] = 4
            ip_part = ip.split('.')
            for i in range(0, 4):
                res[i + 28 + self.name_length] = int(ip_part[i])
            return bytes(res)
        else:
            self.RCODE = 5
            res = bytearray(32 + self.name_length)
            res[0] = self.ID >> 8
            res[1] = self.ID % 256
            res[2] = (self.QR << 7) + (self.Opcode << 3) + (self.AA << 2) + (self.TC << 1) + self.RD
            res[3] = (self.RA << 7) + (self.Z << 4) + self.RCODE
            res[4] = self.QDCOUNT >> 8
            res[5] = self.QDCOUNT % 256
            res[6] = self.ANCOUNT >> 8
            res[7] = self.ANCOUNT % 8
            res[8] = self.NSCOUNT >> 8
            res[9] = self.NSCOUNT % 8
            res[10] = self.ARCOUNT >> 8
            res[11] = self.ARCOUNT % 8
            for i in range(12, 16 + self.name_length):
                res[i] = self.data[i]
            res[16 + self.name_length] = 0xc0
            res[17 + self.name_length] = 0x0c
            res[18 + self.name_length] = 0
            res[19 + self.name_length] = 1
            res[20 + self.name_length] = 0
            res[21 + self.name_length] = 1
            for i in range(22 + self.name_length, 26 + self.name_length):
                res[i] = 0
            res[26 + self.name_length] = 0
            res[27 + self.name_length] = 4
            ip_part = ip.split('.')
            for i in range(0, 4):
                res[i + 28 + self.name_length] = int(ip_part[i])
            return bytes(res)
```

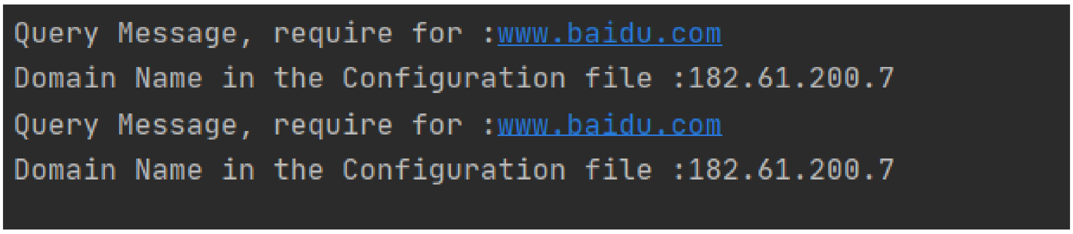
测试结果

- 访问`www.baidu.com`
    - `Powershell`控制台`nslookup`输出

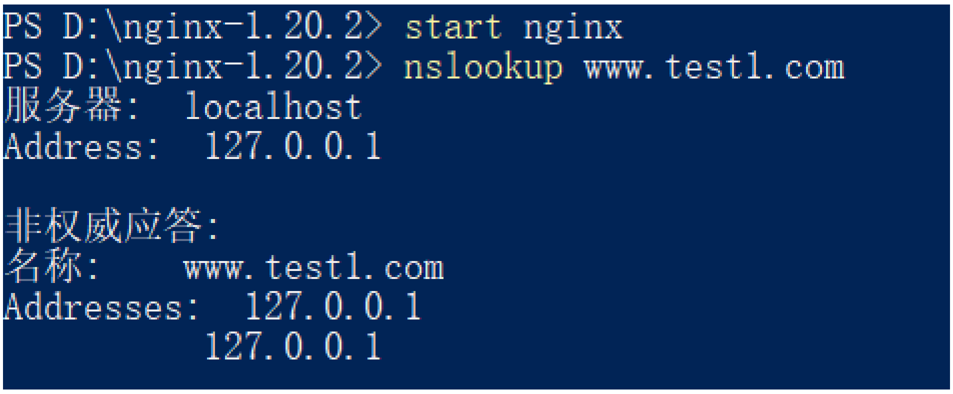
      ```
      PS D:\nginx-1.20.2> nslookup www.baidu.com
      服务器:  localhost
      Address:  127.0.0.1

      非权威应答:
      名称:    www.baidu.com
      Addresses:  182.61.200.7
                182.61.200.7
      ```

    - `Python`控制台输出

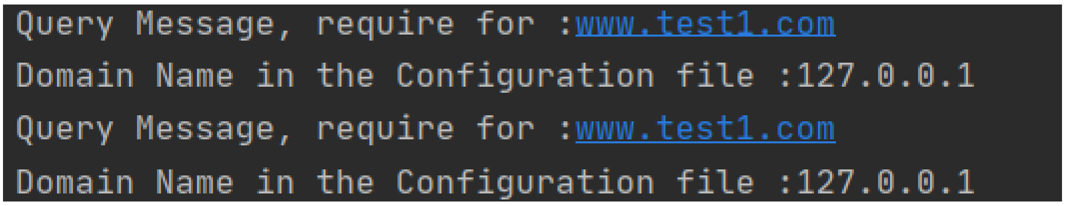
      ```
      Query Message, require for :www.baidu.com
      Domain Name in the Configuration file :182.61.200.7
      Query Message, require for :www.baidu.com
      Domain Name in the Configuration file :182.61.200.7
      ```

- 访问`www.test1.com`
    - `Powershell`控制台`nslookup`输出



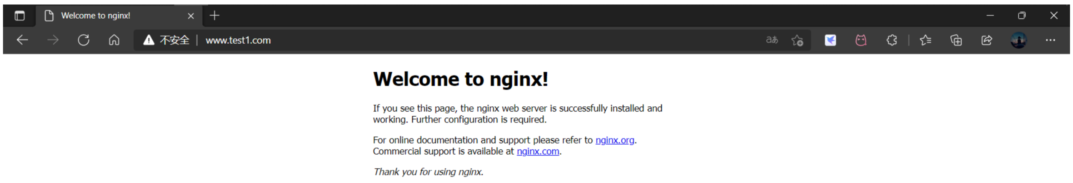
```
PS D:\nginx-1.20.2> start nginx
PS D:\nginx-1.20.2> nslookup www.test1.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    www.test1.com
Addresses:  127.0.0.1
          127.0.0.1
```

  - `Python`控制台输出



```
Query Message, require for :www.test1.com
Domain Name in the Configuration file :127.0.0.1
Query Message, require for :www.test1.com
Domain Name in the Configuration file :127.0.0.1
```

  - 浏览器输出



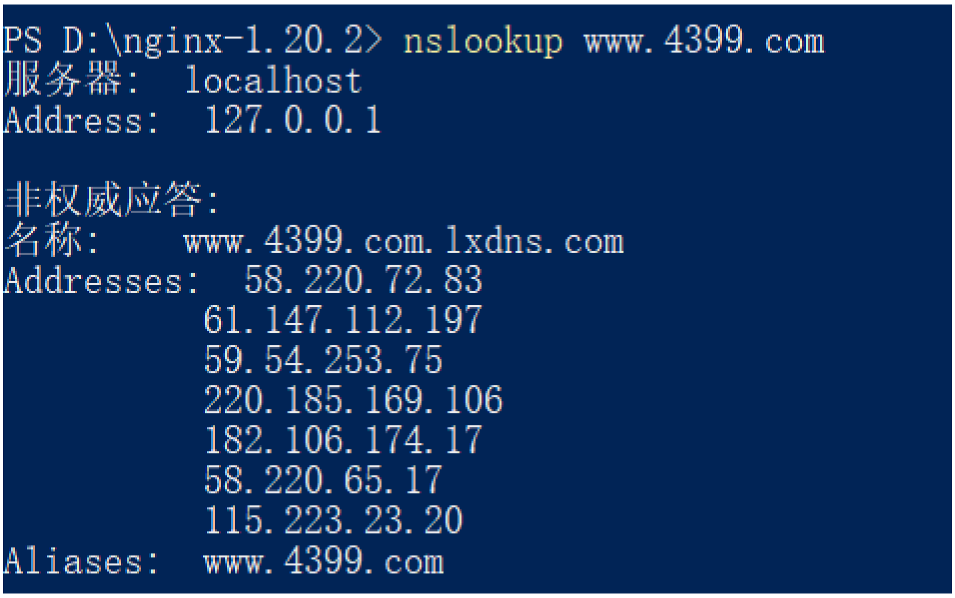
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

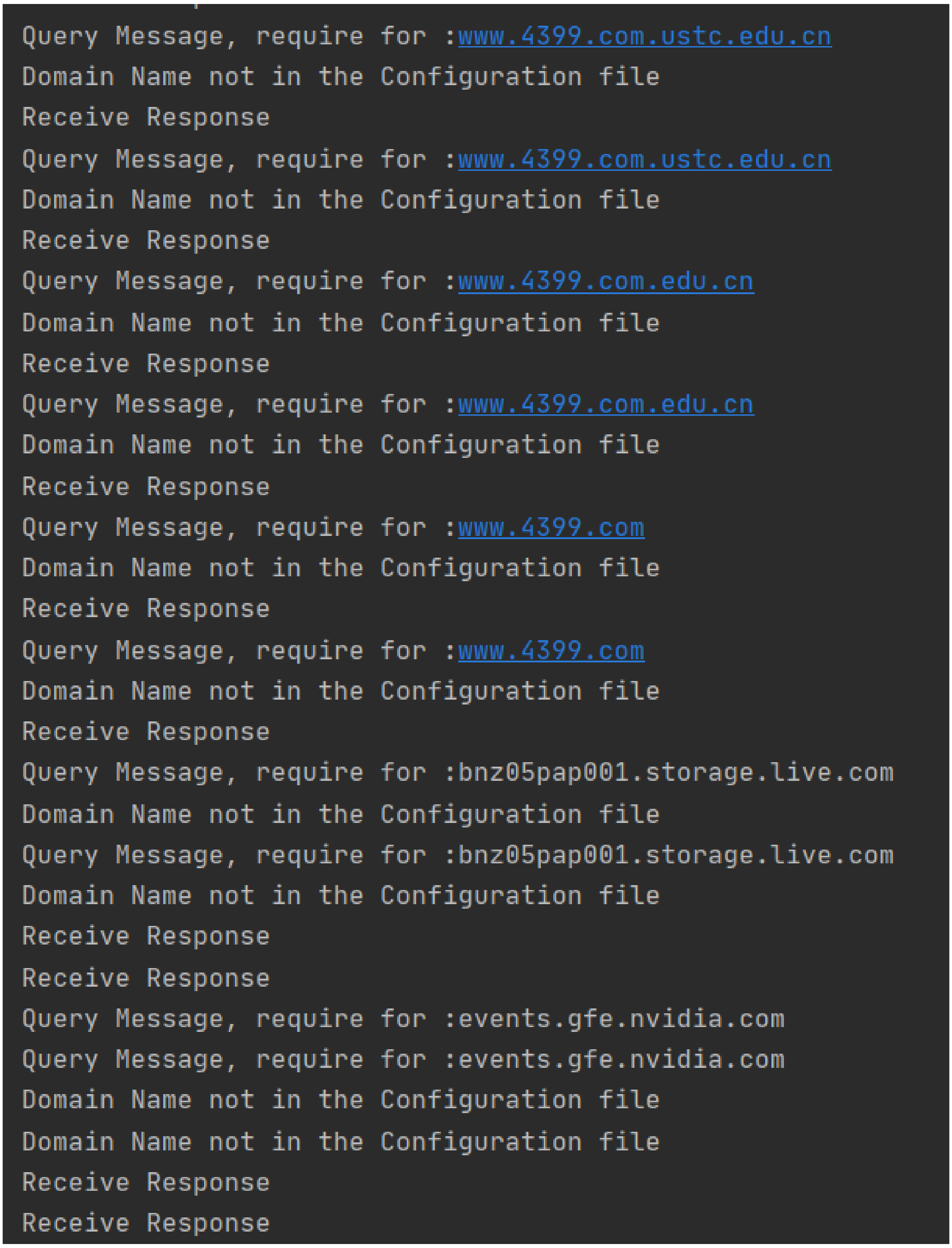
Thank you for using nginx.

- 访问`www.4399.com`
  - `Powershell`控制台`nslookup`输出



```
PS D:\nginx-1.20.2> nslookup www.4399.com
服务器:  localhost
Address:  127.0.0.1

非权威应答:
名称:    www.4399.com.lxdns.com
Addresses:  58.220.72.83
          61.147.112.197
          59.54.253.75
          220.185.169.106
          182.106.174.17
          58.220.65.17
          115.223.23.20
Aliases:  www.4399.com
```

  - `Python`控制台输出



```
Query Message, require for :www.4399.com.ustc.edu.cn
Domain Name not in the Configuration file
Receive Response
Query Message, require for :www.4399.com.ustc.edu.cn
Domain Name not in the Configuration file
Receive Response
Query Message, require for :www.4399.com.edu.cn
Domain Name not in the Configuration file
Receive Response
Query Message, require for :www.4399.com.edu.cn
Domain Name not in the Configuration file
Receive Response
Query Message, require for :www.4399.com
Domain Name not in the Configuration file
Receive Response
Query Message, require for :www.4399.com
Domain Name not in the Configuration file
Receive Response
Query Message, require for :bnz05pap001.storage.live.com
Domain Name not in the Configuration file
Query Message, require for :bnz05pap001.storage.live.com
Domain Name not in the Configuration file
Receive Response
Receive Response
Query Message, require for :events.gfe.nvidia.com
Query Message, require for :events.gfe.nvidia.com
Domain Name not in the Configuration file
Domain Name not in the Configuration file
Receive Response
Receive Response
```