Computer Network lab3 实验报告

姓名 曾闻天

学号 231880147

邮箱 2249428377@qq.com

完成日期 2025.5.18

Computer Network lab3 实验报告
1.实验名称
Lab 3: Respond to ARP
2.实验目的
3.实验进行
3.1 Preparation
3.2 Handle ARP Requests
3.3 Cached ARP Table
3.4 测试与部署
4.实验总结

1 实验名称

Lab 3: Respond to ARP

2 实验目的

1. 理解并实现地址解析协议（ARP）的基本功能。
2. 学习如何在路由器中响应 ARP 请求，获取目标设备的 MAC 地址。
3. 构建并维护 ARP 缓存表，实现对已解析地址的快速查找。
4. 掌握路由器数据包的接收与转发机制。

3 实验进行

3.1 Preparation

下载实验模板代码，初始化项目。
使用 Switchyard 框架搭建基本的路由器框架。

3.2 Handle ARP Requests

实现对 ARP 请求的处理逻辑：

判断接收到的报文是否为 ARP 请求。
检查目标 IP 地址是否为路由器接口的 IP 地址。
如果匹配，则生成并发送 ARP 回复。
如果不匹配，则丢弃该请求。

```
def longest_prefix_match(self, ipaddr: switchyard.llnetbase.IPv4Address):
def handle_packet(self, recv: switchyard.llnetbase.ReceivedPacket):
        timestamp, ifaceName, packet = recv
        # TODO: your logic here
        if 'Arp' not in packet.headers():
            log_info(f"No ARP header! Ignore it")
            return
        else:
            arp = packet.get_header(Arp)

        self.table.add(arp.senderprotoaddr, arp.senderhwaddr)

        try:
            targethwaddr =
self.net.interface_by_ipaddr(arp.targetprotoaddr).ethaddr
        except KeyError:
            log_info(f"No interface assigned to {arp.targetprotoaddr}! Ignore
it")
            return

        self.net.send_packet(ifaceName, create_ip_arp_reply(targethwaddr,
arp.senderhwaddr, arp.targetprotoaddr, arp.senderprotoaddr))
        log_info(f"Send ARP reply asking {arp.targetprotoaddr}")
```

3.3 Cached ARP Table

实现 ARP 缓存表，记录 IP 地址与 MAC 地址的映射关系。
缓存表支持超时机制，超时时间为 10 秒。

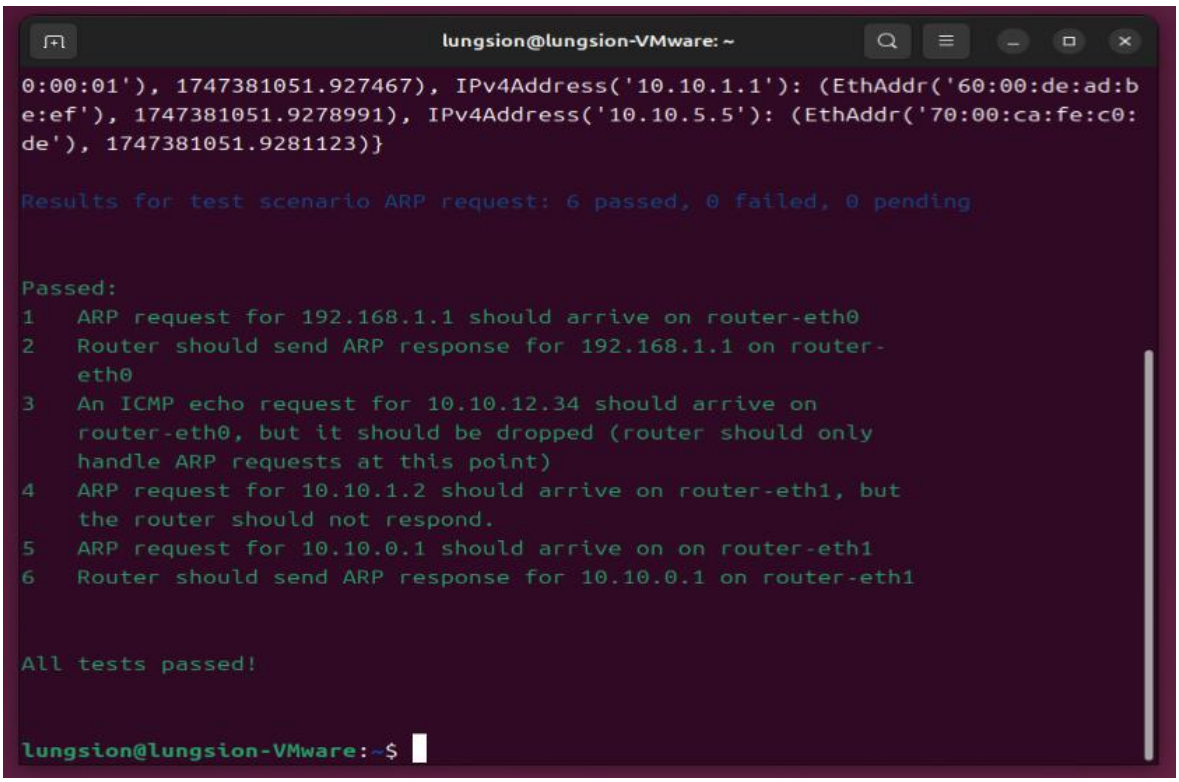
```
class CachedTable(object):
    def __init__(self):
        self.__table = {}

    def isintable(self, ipaddr):
        for i, j in self.__table.items():
            if i == ipaddr:
                if time.time() - j[1] > 10:
                    del self.__table[i]
                    return False
                else:
                    return True
```

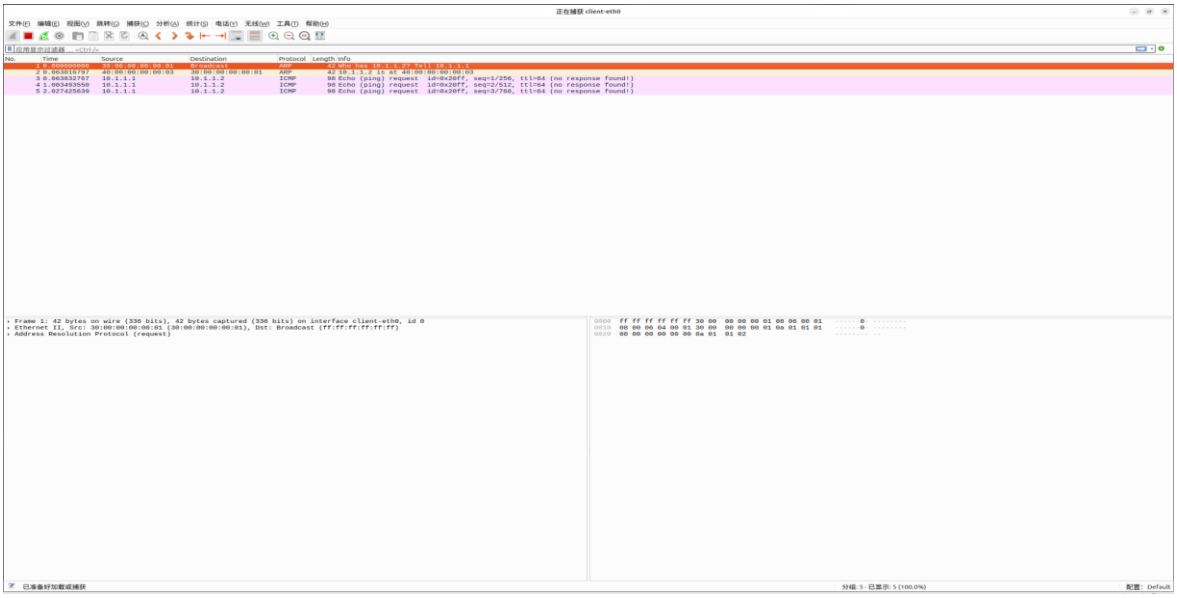
```
        return False

    def add(self, ipaddr, macaddr):
        self.__table[ipaddr] = (macaddr, time.time())

    def get(self, ipaddr):
        return self.__table[ipaddr][0]

    def print(self):
        log_info(f"{self.__table}")
```

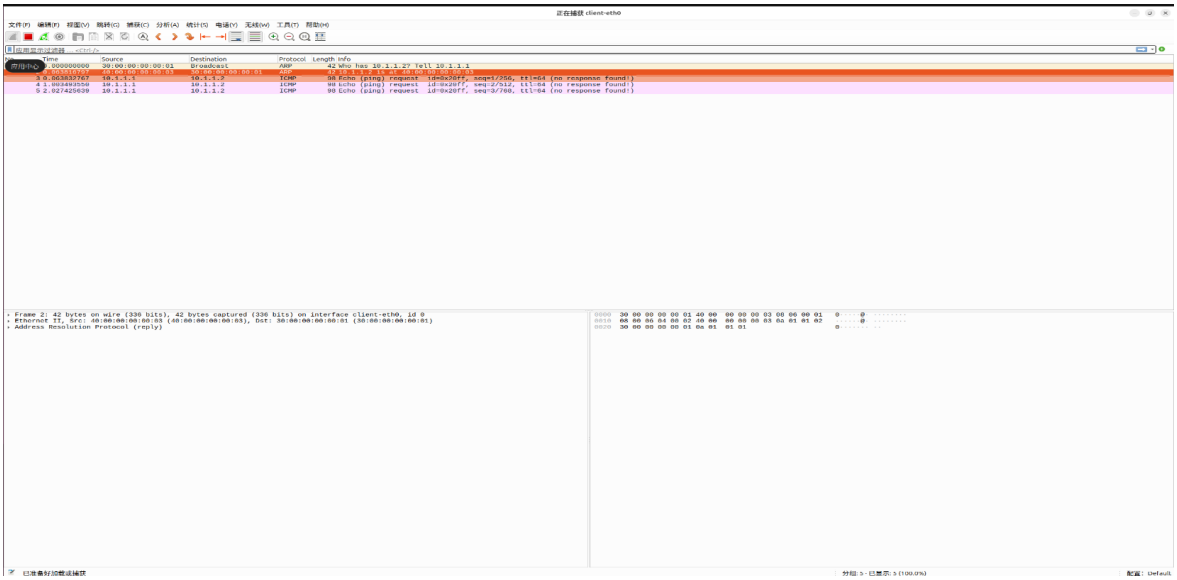
3.4 测试与部署

使用 Switchyard 提供的测试用例进行功能测试。

在 Mininet 环境中部署路由器，使用 Wireshark 捕获并分析 ARP 请求与响应数据包。根据 ARP 缓存获取下







4 实验总结

通过本次实验，我深入理解了地址解析协议（ARP）的工作原理，掌握了如何在路由器中实现 ARP 请求的响应与缓存管理。实验过程中，我学习了以下内容：

ARP 协议的基本原理：了解了 ARP 请求与响应的格式，以及如何通过 ARP 协议获取目标设备的 MAC 地址。

路由器数据包处理机制：掌握了路由器如何接收、解析并转发数据包。

缓存表的实现与管理：学会了如何设计并实现 ARP 缓存表，以及如何通过超时机制优化缓存性能。

Mininet 与 Wireshark 的使用：熟悉了 Mininet 网络仿真环境和 Wireshark 网络分析工具的使用方法。

本次实验为后续学习动态路由协议（如 RIP、OSPF）奠定了基础，使我更加深入地理解了计算机网络中路由器的工作机制。

致谢 在此,我向对本报告的工作给予支持和建议的老师和助教,南京大学南京大学智能软件与工程学院殷亚凤教授及其领导的助教们表示感谢.