南京大学本科生实验报告

课程名称: 计算机网络
任课教师: 黄程远
助教: 刘松岳
学院: 计算机学院
学号: 231220095
专业（方向）: 计算机科学与技术
姓名: 黄晓坛
Email: 231220095@smail.nju.edu.cn
开始/完成日期: 11.2/11.7

1. 实验名称

Lab 4: Forwarding Packets

2. 实验内容

本实验的目标是实现路由器对 ARP 请求的响应，并进行基本的 IP 转发和 ARP 请求发送功能。在实验中，完成了如下几个主要部分：

Cached ARP Table

- **获取路由器端口的 IP 和 MAC 地址**：在路由器的构造函数中，通过 `net.interfaces()` 获取路由器所有端口的 IP 和 MAC 地址信息，并存储到一个字典中，方便后续查找和快速构建 ARP 回复。

Handle ARP Request

- **处理 ARP 请求**：在 `handle_packet` 方法中，判断接收到的包是否是 ARP 请求。如果请求包的目标 IP 地址是路由器某个端口的 IP 地址，则说明这是一个发向路由器的 ARP 请求。根据之前缓存的接口信息，构造 ARP 回复包并发送。

Forwarding IP Packet and ARP Requests

- **处理 IP 转发与 ARP 请求发送**：在确定目标 IP 地址后，通过查找路由表确定下一跳 IP。如果目标地址不在 ARP 缓存中，则发送 ARP 请求来解析 MAC 地址，并将数据包加入待处理队列，等待 ARP 响应。每隔 1 秒重试一次 ARP 请求，最多发送 5 次。若未收到 ARP 回复，则丢弃数据包。

3. 实验结果

3.1 匹配目标 IP 地址

在 `lookup_forwarding_table` 方法中，利用最长前缀匹配算法来确定与目标 IP 地址最匹配的路由条目。具体步骤如下：

1. **解析转发表**：通过读取文件 `forwarding_table.txt` 和路由器接口的网络信息，构建包含网络地址、子网掩码、下一跳地址及接口名称的转发表。
2. **最长前缀匹配**：对于每一个接收到的 IP 数据包，提取其目标 IP 地址 `dest_ip`，遍历转发表中的每一条记录，比较目标地址和表中网络地址的前缀长度，选择前缀匹配最长的条目作为最终转发目标。具体代码逻辑如下：

```
def lookup_forwarding_table(self, dest_ip: IPv4Address):
    best_match = None
    longest_prefix = -1
    for network, netmask, next_hop, interface in self.forwarding_table:
        prefix_length = IPv4Network(f"{network}/{netmask}").prefixlen
        if (int(dest_ip) & int(netmask)) == int(network):
            if prefix_length > longest_prefix:
                best_match = (network, netmask, next_hop, interface)
                longest_prefix = prefix_length
    return best_match
```

通过这种方式，路由器能够实现精确匹配，保证数据包被转发至最佳下一跳。

3.2 处理数据包转发和 ARP 请求生成

在 IP 包转发过程中，路由器需要根据下一跳地址进行 MAC 地址解析，并在必要时生成 ARP 请求。处理流程如下：

1. **检查 ARP 缓存**：在确定下一跳后，首先在 ARP 缓存中查找对应的 MAC 地址。如果找到缓存的地址，则直接构建以太网头并转发数据包。
2. **生成 ARP 请求**：如果未在 ARP 缓存中找到 MAC 地址，则生成 ARP 请求数据包并将该 IP 数

据包放入待处理队列中，同时记录请求的发送时间和重试次数：

```
def add_packet(self, packet, next_hop, out_iface):
    current_time = time.time()
    pending_packet = PendingPacket(packet, next_hop, out_iface, current_time)
    self.queue.append(pending_packet)
    self.send_arp_request(next_hop, out_iface)
```

3. **定期重试**:每秒检查队列中的待处理数据包，如果超过 1 秒未收到 ARP 响应，则重发 ARP 请求，最多重试 5 次。若仍未收到回复，则丢弃该数据包：

```
def retry_pending_packets(self):
    current_time = time.time()
    for packet_info in list(self.queue):
        if current_time - packet_info.time_sent >= self.retry_interval:
            if packet_info.retries < self.max_retries:
                self.send_arp_request(packet_info.next_hop, packet_info.out_ifac
                packet_info.retries += 1
                packet_info.time_sent = current_time
                log_info(f"重新发送ARP请求以解析 {packet_info.next_hop}，重试次数: {p
            else:
                log_info(f"达到最大重试次数，丢弃数据包，目标地址: {packet_info.next_h
                self.queue.remove(packet_info)
```

4. 实验总结

经过这次试验对路由器转发原理、arp协议等有了更深层的理解。