Report

任家纬

2020012377

解题思路

本次实验要实现的有三个函数。初步观察下，查看路由表的函数`lookup`最容易实现，且应会被`handlePacket`调用，`periodicCheckArpRequestsAndCacheEntries`则被另外的线程独立调用。

因此，首先考虑实现`lookup`函数，其次实现核心函数`handlePacket`，最后再实现`periodicCheckArpRequestsAndCacheEntries`

lookup

```
/ * *
* This method should lookup a proper en try in the rou t ing tab le
* using " longes t – p r e f i x match" algorithm
* /
RoutingTableEntry
RoutingTable : : lookup (uint32_t ip) const ;
```

- `RoutingTable`类中的转发表是一个列表，因此只需遍历列表使用最长前缀匹配即可
- ip地址与子网掩码mask按位与后与表项匹配
- 最长前缀匹配

  > 不妨设两个32位数为a、b。a ^ b必然会得到形如“00……1xx……”的数，其中前面的0可以没有。两者匹配的最长前缀实质上就是从左到右第一个1左边的0的个数。

- 匹配成功返回转发表项，失败抛出`runtime_error`

handlePacket

```
/**
   * IMPLEMENT THIS METHOD
   *
   * This method is called each time the router receives a packet on
   * the interface.  The packet buffer \p packet and the receiving
   * interface \p inIface are passed in as parameters. The packet is
   * complete with ethernet headers.
*/
void
handlePacket(const Buffer& packet, const std::string& inIface);
```

1. 首先解析传入的packet，得到以太帧首部。若目的mac地址既不是路由器对应接口的mac地址，也不是广播地址，直接返回；否则，进行下一步
2. 检查以太帧首部的type，若是0x0806，转去处理ARP；若是0x0800，转去处理IPV4
3. 处理ARP

1. 解析以太帧的payload，获取ARP的首部。若hardware type不是0x0001，直接返回
2. 检查opcode，看是ARP请求还是ARP回复
3. 若是ARP请求，首部的目的ip应当是路由器接口的ip，否则直接返回
4. 发送响应分组，目的mac是发来的地址，源mac是接口mac地址，目的ip是发来ip，源ip是接口ip
5. 若是ARP回复，在ARP表中新增一项，ip和mac是发来的包的源ip和mac
6. 将所有正在排队的包发出

4. 处理IPV4
    1. 解析ip首部和icmp首部
    2. 检验包的长度和检验和，检验和可以使用内置函数
    3. 在ARP表中查看有无已存在的表项，没有则加入一条
    4. 超时的信息要发一个返回的icmp，type11、code0
    5. unreachable的信息要发一个返回的icmp，type3、code3
    6. 目的为路由的包，返回一个icmp，type0、code0
    7. 要转发的包，先利用刚实现的`lookup`函数查看转发表，匹配失败则直接返回
    8. 修改以太帧首部的源mac为路由器上将要发包的端口mac。ip首部ttl减1，重新计算检验和。
    9. 如果不知道目的mac，则先把包加入队列等待广播获取目的mac；若知道目的mac，直接发包

periodicCheckArpRequestsAndCacheEntries

```
/**
   * IMPLEMENT THIS METHOD
   *
   * This method gets called every second. For each request sent out,
   * you should keep checking whether to resend a request or remove it.
   *
   * Your implementation should follow the following logic
   *
   *     for each request in queued requests:
   *         handleRequest(request)
   *
   *     for each cache entry in entries:
   *         if not entry->isValid
   *             record entry for removal
   *     remove all entries marked for removal
*/
void
periodicCheckArpRequestsAndCacheEntries();
```

- 根据提示，对于所有在队列中的请求，若发送5次以上，移除；否则将其再次发出
- 对于所有在ARP表中的条目，如果被设为无效了，移除

困难

- 环境搭建。本次实验采用ubuntu16.04，且不能wsl，因此花了大力气来配一个虚拟机
- 数据包中的数值是大端存储，机器内是小端存储，不能直接比较，需要转换
- 解析包头可以利用结构指针的强制转换，这会大大减轻工作量
- 实验时发现ping路由器无效，发现发包时包的大小设错了。原来的包的数据应该也要发回去。

建议

- 助教团队可以提前把实验环境配置好。可以提供在目前主流环境（ubuntu20，wsl）下的实验方法，或是把带有ubuntu16的虚拟机直接打包提供给学生