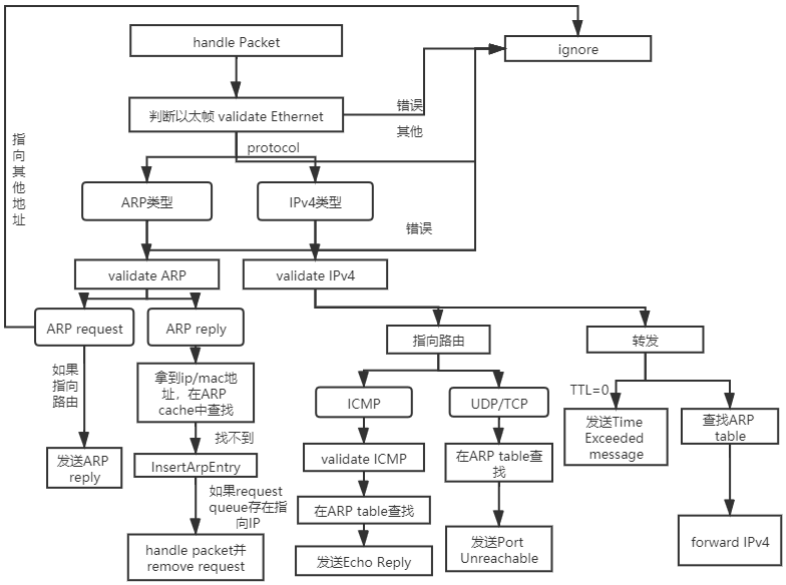
Simple Router 文档

2018013361 余齐齐

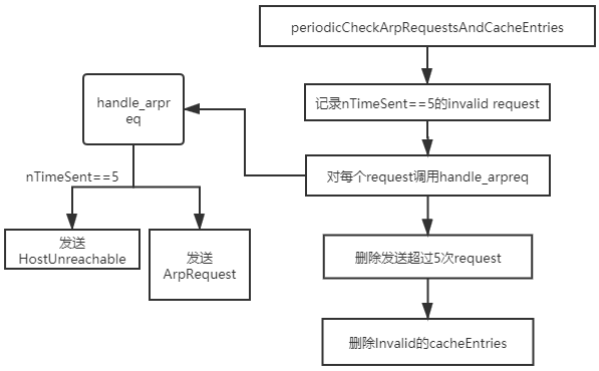
项目流程

项目基本流程如下：

- simple-router.handlePacket()



- Arp-cache.periodicCheckArpRequestsAndCacheEntries()



首先对以太帧进行检测：packet的大小、Ether_type、Ether_dhost应该是router接口的MAC地址或广播地址。

然后分类型讨论（ARP或IPv4）

- 如果是ARP：

  首先对ARP包进行检测：packet的大小、hardware Type、Protocol Type、HW addr len、Port addr len、Opcode.

  ARP有两种类型：

  - ARP request

    如果request的ip是router接口的ip地址的话，发送ARP reply，否则ignore。

    发送ARP reply。

- ARP reply

  需要先拿到arp Reply对应的ip和mac对应地址，拿到对应地址后，需要在arp cache中查找，如果查找不到就需要insertArpEntry。

  insertArpEntry:在request queue中查找此IP。

  如果存在的话，就handlePacket并removeRequest。

- 如果是IPv4

  首先对IPv4数据报进行检验：数据报的大小、checksum。

  对不同类型讨论：

  （1）指向路由器。

  （2）指向其他地址，需要转发。

  - 指向路由器
    - ICMP

      首先判断类型是否有错：长度、类型、checksum。

      然后调用routingTable.lookup，在ARP table中lookup，查找目的IP对应的MAC地址。如果arp_table里没有，则queueRequest。

      如果有的话则发送echo reply。

    - UDP/TCP

      调用routingTable.lookup，在ARP table中lookup，查找目的IP对应的MAC地址。如果arp_table里没有，则queueRequest。

      如果查找到，则发送Port Unreachable message.

  - 如果指向其他地址
    - 如果TTL==0

      发送 Time Exceeded message.

    - 否则转发

      调用routingTable.lookup，在ARP table中lookup，查找目的IP对应的MAC地址。如果arp_table里没有，则queueRequest。

      如果查找到，调用forwardIPv4.

- 在Arp-cache中

  记录nTimeSent==5的request。

  对每个request调用handle_arpreq，在此函数中对request进行处理。当request->nTimesSent==5时，返回HostUnreachable；其他发送ArpRequest。在对Arp Reply进行处理的时候，

  删除发送超过5次的request。

  删除Invaild的cacheEntries。

遇到的问题

- 我花了将近3-4天的时间阅读助教的文档来理清思路、理解整个的设计流程。在理解设计流程之后，代码编写就变得较为轻松了。
- 在编写代码和测试Debug时，我遇到的问题主要有：
  - 对htons,htonl,ntohs,ntohl等理解不清。

    htons,htonl,ntohs,ntohl等函数涉及到了主机顺序码到网络顺序码的转换。

    当我从网络上收到packet的时候并需要对字段进行处理时，就需要进行从网络顺序码到主机顺序码的转换。当我要向网络发送数据报时，需要进行从主机顺序码到网络顺序码的转换。

    对于uint_16(unsigned short)需要进行htons/ntohs，对uint_32(unsigned int)需要进行ntohl/htonl.但对uint_8(unsigned char)等不应该使用转换函数，因为只有一字节，如果使用，会导致错误。
  - 对ARP request的具体数据理解不清。

    ARP request的Ether_dhost是FF:FF:FF:FF:FF:FF，Arp_tha是00:00:00:00:00:00，因此在回复时，需要将shost和sha变为iface->addr.data().
  - 对checksum的理解不清。

    checksum的两种使用方式：

    1.计算

    方法：

    - hdr->sum置0
    - 调用cksum
    - hdr->sum=计算结果

    2.验证

    方法：直接cksum，算出来是0xffff是对的，其他则错误。
  - 在传输较大文件时（100M），由于在simple-router中查找的ARP Table，导致锁了多次，造成了死锁，出现了堵塞。

    对于在传输较大文件时出现的停滞，有两种情况：1.TCP拥塞控制，2.死锁。

    将锁由mutex改成了recursive mutex类型，就可以正常的传输，解决了死锁问题。

感想

通过这次实验我对路由器的工作原理有了更多的了解，对ARP、ICMP、Ethernet、IPv4数据报的形式更加清晰。

参考

参考了https://github.com/hoooga/Simple_Router，这份代码对我理解路由原理有很大帮助，十分感谢！