计网 LAB4 实验报告

姓名：徐佳美　　学号：181860117
专业：计算机科学与技术系
任课老师：李文中
邮箱：181860117@smail.nju.edu.cn
开始/完成日期：4.17-4.20

一．实验名称：
包的转发

二．实验目的：
接收和转发到其他主机的包，学习最长前缀匹配的使用。了解 IPv4 和 ARP 包头的使用和构造。

三．实验内容和核心代码：

$task2$:

（1） 转发表的构造：
函数：$constructTable$
转发表：fowardTable[]
表项：子网　前缀长度　下一跳　　端口名

1. 从 forwarding_table.txt 按行读入：
用 strip 和 split 分割每一项，前缀和掩码合并成子网地址，计算前缀长度，依次存入；
2. 路由器的端口：
$intf.ipinterface.network$获取子网，存入表格，$prefixlen$获取长度，下一跳暂设为 none；

```
def constructTable(net):
    for rawline in open("forwarding_table.txt","r"): #设置文件对象并读取每一行文件
        if not rawline :
            break
        line = rawline.strip()
        if not line:
            continue
        (prefix, mask, nextHop, port) = line.split(' ')
        temp=[]

        netaddr=prefix+'/'+mask
        netaddr=IPv4Network(netaddr) #entry:netaddr,length,nextHop,port
        temp.append(netaddr)
        temp.append(netaddr.prefixlen)
        temp.append(IPv4Address(nextHop))
        temp.append(port)
        fowardTable.append(temp)
```

```
    for intf in net.interfaces():    #the router intf
        temp=[]
        netaddr=intf.ipinterface.network
        temp.append(netaddr)
        temp.append(netaddr.prefixlen)
        temp.append("none")
        temp.append(intf.name)
        fowardTable.append(temp)
```

（2） 最长前缀匹配：

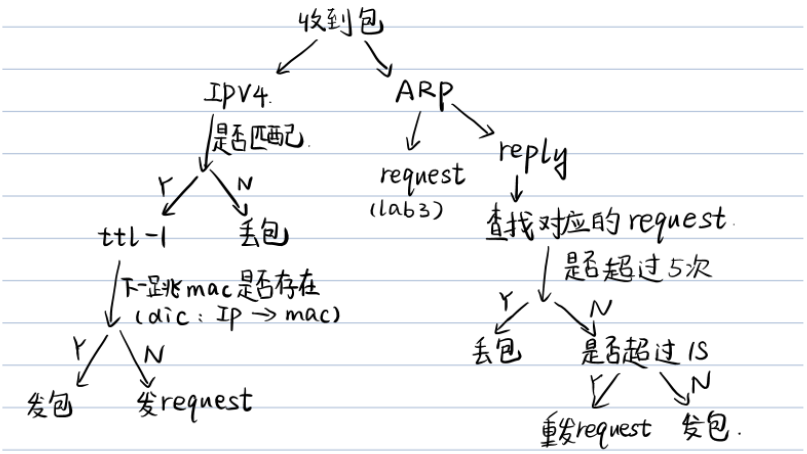
函数：longmatch(destaddr):

传入目的地址，返回在转发表中匹配的表项下标，若为-1，则无匹配；匹配：检查目的地址是否在表项子网中，若在，比较长度，选择最长匹配项的下标；

```
def longmatch(destaddr):
    length=len(fowardTable)
    index=-1  # return the index of match entry
    leng=0
    for i in range(length):
        prefixnet=fowardTable[i][0]
        matches = destaddr in prefixnet
        if matches==True and fowardTable[i][1]>leng :
            index=i
            leng=fowardTable[i][1]
    return index
```

task3:

（1）流程图：



（2）数据结构：

dic:存储 ip 对应的 mac 地址
$arpTable$:缓存发出去的$arp\ request$:含 arprqt ,上次发送时间，发送次数，端口名
dq:缓存没发出去的包；含 arprqt,pkt,端口名

（3） 具体实现：

1. 发现一个没有目的地 mac 无法发送的包:
   构造 arprqt 包，调用 ifNewEntry 函数：检查是否是缓存过这个 request；若无，在 arpTable 和 dq 中添加新表项，若有，检查是否在 dq 中缓存过这个包，若无，添加新表项；

```
def ifNewEntry(arprqt,pkt,portname,self):
    hasIn1=False
    hasIn2=False
    len1=len(arpTable)
    for i in range (len1):
        if arpTable[i][0]==arprqt : # if had cache the arprqt,check if had cache
the pkt
            hadIn1=True
            len2=len(dq)
            for j in range(len2):
                if dq[j][0]==arpTable[i][0]:
                    hasIn2=True
                    break
                if hasIn2==False : # cache the pkt
                    temp=[]
                    temp.append(arprqt)
                    temp.append(pkt)
                    temp.append(portname)
                    dq.append(temp)
            break
    if hasIn1==False:
        #cache the arprqt
        temp1=[]
        temp1.append(arprqt)
        temp1.append(time.time())
        self.net.send_packet(portname,arprqt)    #send request
        temp1.append(1)
        temp1.append(portname)
        arpTable.append(temp1)
        #cache the pkt
        temp=[]
        temp.append(arprqt)
        temp.append(pkt)
```

```
        temp.append(portname)
        dq.append(temp)
```

2. Arpquery(self,mac)：

如果 reply 有效，更新 dic,并将正确的 mac 传入，调用这个函数；
或者在 while 循环开始时调用；

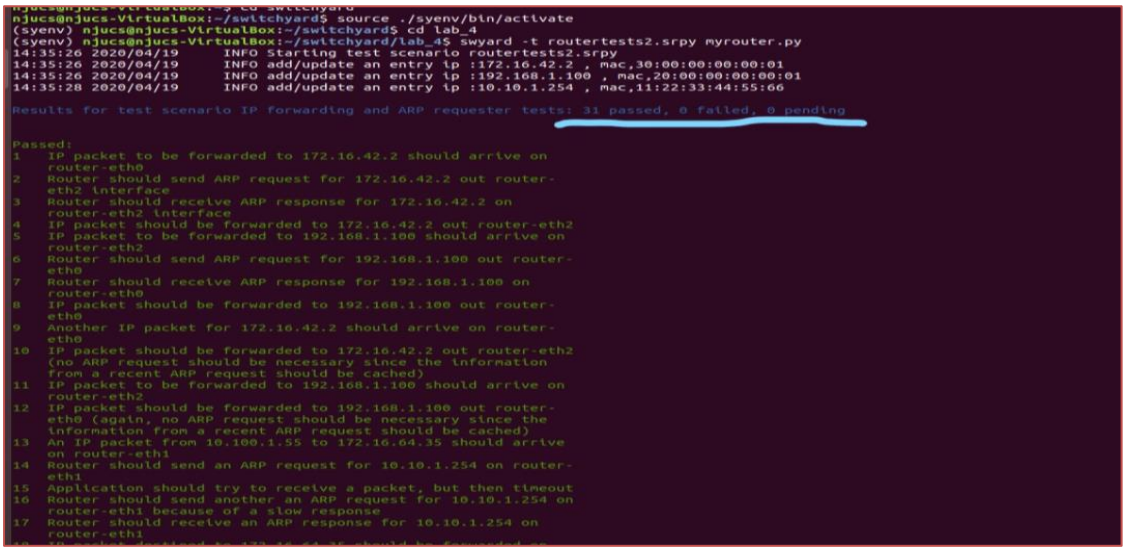
①检查是否有 arpTable 的发送次数为-1，（收到正确 reply 的标志），如有，在 dq 中查找此 arp request 对应的包，将他们都发出去；
发完后，删除这个 request 和 request 对应的包；
②检查是否有 arpTable 的发送次数为 5，若有，说明已经长时间无应答，丢包：
在 arpTable 中删除这个表项，在 dq 中删除这个 request 对应的包；
③发现一个 request 发送未超过 5 次，且超过 1 秒无应答：重新发送一次 request;

```
def arpquery(self,mac):
    len1=len(arpTable)
    #check if there is a tag,send the pkt and then del the entry
    for i in range (len1):
        if arpTable[i][2]==-1:
            len2=len(dq)
            for j in range(len2):
                if dq[j][0]==arpTable[i][0]:
                    pkt=dq[j][1]
                    pkt[Ethernet].dst=mac
                    self.net.send_packet(dq[j][2],pkt)
      else:    # no response
            if arpTable[i][2]<5:
                if time.time()-arpTable[i][1]>=1:    #has pass 1s,send again
                    self.net.send_packet(arpTable[i][3],arpTable[i][0])
                    arpTable[i][1]=time.time()
                    arpTable[i][2]+=1   #count++
    #del pkts send successfully and arprqt

    for i in range (len1-1,-1,-1):
        if arpTable[i][2]==-1 or arpTable[i][2]>=5:
            len2=len(dq)
            for j in range(len2-1,-1,-1):
                if dq[j][0]==arpTable[i][0]:
                    del dq[j]
            del arpTable[i]
```

四．测试与验证：

（1）测试结果：



（2）mininet 验证：

在 server1 节点 ping client： ping -c2 10.1.1.1

第一个 arp，server1 不知道 client 的 mac，发出一个 request;

第二个 arp，client 向 server1 发出一个 reply;

第三行和第五行：server1 向 client 发的两个包；

第四行和第六行：client 回复 server1 自己收到包。