**网络协议栈分析与设计课程作业**

**DSR路由协议代码分析**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学号 | 姓名 | 班级 | 负责模块 | 成绩 |
| 201693125 | 张璐鹏 | 软网1602 | 1. 协议简介 2. 代码介绍   3. 路由请求  4. 协议基本原理分析  5. 协议特点 |  |
| 201692158 | 许艺严 | 软网 1601 | 1. 引言  2. 路由应答  3. 协议基本结构分析  4. 数据结构  5. 文献搜集 |  |

大连理工大学

Dalian University of Technology

目 录

1 引言 1

2 DSR 路由协议简介 2

2.1 路由发现 3

2.2 路由维护 3

2.3 文件介绍 4

3 DSR结构分析 5

3.1 DSR分组结构 5

3.1.1 dsr\_node路由节点结构 5

3.1.2 dsr\_opt结构 6

3.1.2.1 路由请求选项 7

3.1.2.2 路由应答选项 9

3.1.2.3 ACK请求选项 11

3.1.2.4 ACK选项 13

3.1.2.5源路由选项 14

3.2 rerr选项结构 17

3.3 maint\_buf选项结构 19

4 DSR协议基本原理 20

4.1 路由发现 20

4.1.1路由请求 21

4.1.2重复RREQ检测 22

4.2 路由维护 23

4.3 缓存机制 24

4.3.1 Send-buf TBL 25

4.3.2 Maint-buf TBL 26

4.4 应答机制 27

4.4.1 ack确认处理机制 27

4.4.2 rrep 应答处理机制 28

4.5 错误处理 29

5 DSR协议特点 30

5.1 DSR协议优点 30

5.2 协议缺点 30

6 总结 31

# 1 引言

无线自组网是一种自组织、自管理的网络，其中所有节点都需要同时充当终端和路由器的角色，具有同等的地位。无中心、可快速展开节点、可移动、多跳等特点让无线自组网中的所有设备的地位是平等的，所有节点都具备路由和转发数据的能力，可以随时组成一个小区域通信系统，其设备支持全球的漫游性。

不依赖于任何集中接入点或网络基础设施。对于这种临时动态网络，提供有效的路由协议对于适应主机的移动性和为可能的链路提供可行的数据传输是必不可少的。同时，由于其具有较高的动态拓扑结构，路由算法已成为自组网研究的热点和难点。目前，人们提出了多达10 - 20种移动自组网路由协议，但最基本、最新颖的只有DSR、TORA、AODV、DSDV、CGSR、ABR等几种。根据不同环境的特点，它们都有各自的特点和优点，在任何情况下，没有一种协议是最好的。

在路由协议DSDV、DSR、AODV和TORA是最具代表性的Ad Hoc网络路由协议,和DSR具有完全不同于传统路由协议创建根据特设网络的特点。同时,它是最好的路由协议整体性能。因此，有必要对DSR路由协议进行更深入、更全面的研究。

无线自组网的特点使得WPAN无论是在军事上还是民用领域都有着广阔的应用前景，成为近年来热点课题之一。传统的路由协议RIP、OSPF在无线互联网中不一定适用，因此能够有效的找到节点路由的动态路由协议至关重要。

DSR协议主要由两个重要的机制组成，路由发现和路由引入；

路由发现的作用是使每个节点都能够动态的找到到达其他节点的路径，当某个节点接收到一个路由请求包的时候，对请求包进行处理判断，如果本次路由请求标识与起始地址已经存在于最近可见的请求中，为了防止重复路由请求包，将会丢弃该路由请求包。

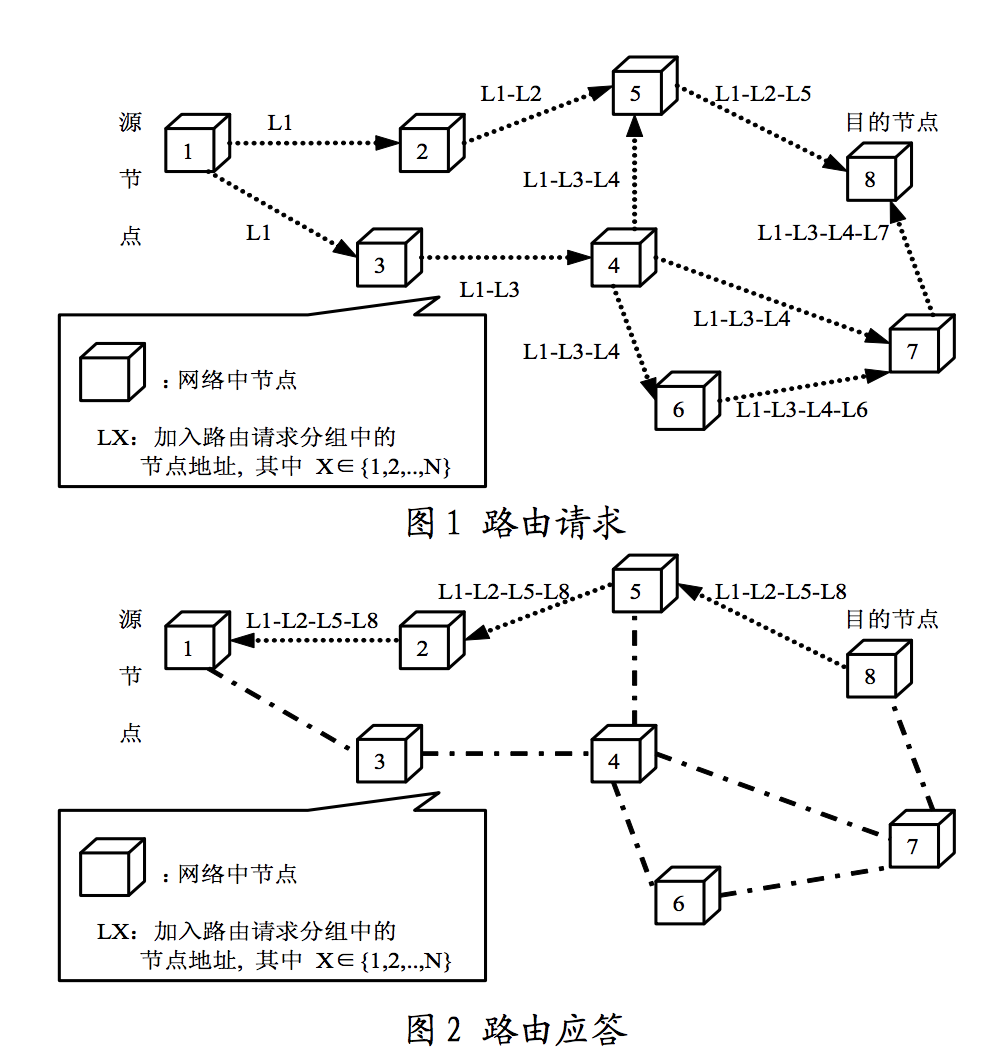
如果地址已经存在于请求的路由记录中，也丢弃路由请求包，以防止产生循环路由。请求包的目标标识与某个节点的地址相吻合，则信息包中的全程路由记录得以确认，返回响应包到起始端，如果请求标识与节点地址不吻合，追加此节点的地址到路由请求包的路由记录中。

DSR协议相比传统路由协议具有明显优势，不需要周期性的广播路由信息，传统路由协议在固定网中往往构造的是双向对称路由。无线移动环境中，DSR可以去除冗余。传统路由协议无法适应网络拓扑动态变化，DSR可以快速适应变化。

综上所述，有必要对DSR路由协议以及它的源码进行深入研究。

# 2 DSR 路由协议简介

动态源路由协议(DSR, Dynamic Source Routing)是一种按需路由协议，允许节点动态发现多个跳转到目标节点路由。所谓的源路由是在每个数据包的头部进行的在到达目标节点(即组中的目标节点)之前必须通过所有组的节点列表包含到目标节点的完整路由。在DSR中，不必周期性的广播路由控制信息，DSR路由协议中的所有状态均为“软状态”，任何状态的丢失都不会影响DSR路由协议的正确操作，因为所有状态都是根据需要创建的，状态丢失后，如果仍需要，它可以容易和快速恢复，从而减少网络带宽开销，避免移动Ad Hoc网络范围大范围路由更新。





## 2.1 路由发现

路由发现的作用是使每个节点都能动态地找到到达其他节点的路径，当某个节点接收到一个路由请求包时，对请求包处理方法有以下４类：

1. 本次路由请求标识与起始端地址已经存在于最近可见的请求中，为防止重复路由请求包，放弃该请求包。
2. 节点地址已经存在于请求的路由记录中，也丢弃路由请求包以防产生循环路由。
3. 请求包的目标标识与某个节点的地址相吻合，则信息包中的全程路由记录得以确认，返回响应包到起始端。
4. 请求标识与节点地址不吻合，追加此节点地 址到路由请求包的路由记录中，并重广播此请求。

当源节点需要与目标节点通信时，它首先查看节点维护的路由缓存，看看是否有到目标节点的路由。如果路由缓存中已经包含到目标节点的有效路由，则立即使用路由数据包;否则，它将向所有邻居广播RREQ(路由请求)包，以启动路由发现进程，查找到目标节点的可用路由。

具体实现会在分析dsr-rreq.h, dsr-rreq.c文件中详细介绍。

## 2.2 路由维护

目的节点收到RREQ后，给源节点返回路由应答（RREP）消息，拷贝RREQ消息中的路由记录。源节点收到RREP后在本地路由缓存中缓存路由信息。

路由维护是运用数据层，点对点的确认来提供丢失的或被破坏的信息包的提早发现和重传输．当 运用源路由发送信息包时，每个传送节点都是依次确认。节点Ｓ向节点Ｄ通过中介节点Ａ，Ｂ，Ｃ发送一个信息包，如果重传次数超过限定值后没有收到接收确认，则该节点返回一个路由错误报文给始发点，收到路由错误报文后，节点路由 缓存器删除该错误的单跳，且所有含有此单跳的路由都必须在这个点上截断。

具体实现会在分析dsr-rrep.h, dsr-rrep.c文件中详细介绍。

## 文件介绍

|  |  |
| --- | --- |
| 文件名 | 说明 |
| dsr.h  dsr-srt.h  dsr-srt.c  dsr-rerr.h  dsr-rerr.c  dsr-module.c  dev.c  dsr-opt.h  dsr-opt.c  dsr-ack.h  dsr-ack.c  dsr-io.c  dsr-pkt.h  dsr-pkt.c  dsr-rrep.h  dsr-rrep.c  dsr-rreq.h  dsr-rreq.c  dsr-rtc.h  dsr-rtc-simple.c  link-cache.c  maint-buf.c  neigh.c  send-buf.c | dsr\_node定义，结构，初始化，配置变量  源路由选项结构，源路由结构  增加，新建源路由节点  路由出错报文，错误分组。  错误分组选项结构，  接收读写配置程序  dsr的网络设备操纵  dsr选项、选项首部  增删改查dsr选项  ack请求选项以及ack选项的构成  对ack选项的操纵  对dsr选项的IO处理  产生内部的dsr pkt  pkt以及pkt opt的产生与操纵  rrep路由应答结构与操作  对rrep包和table的操作  rreq请求的opt的结构  rreq路由请求  DSR route cache API 查增删刷新  rtc entry结构  lc的node link 增删改查  用于处理pre hop -> next hop之间时的信息传送所用buf  邻节点的结构、增删改查、系统崩溃恢复时间，邻节点表的维护  增删改查send buf |

# 3 DSR结构分析

## 屏幕快照%202018-12-13%20193.1 DSR分组结构

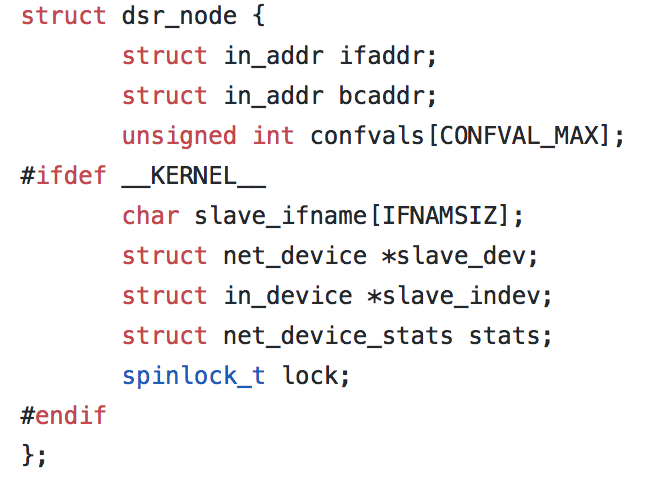


DSR协议使用一个结构特殊的DSR头来携带控制信息， 这个DSR头可以被任何已经存在的IP分组携带。DSR选项头的种类包括路由请求、路由应答、ACK请求、ACK、源路由。

### 3.1.1 dsr\_node路由节点结构

struct dsr\_node如下截图所示：

dsr.h



dsr.h

图6 struct dsr\_node的定义



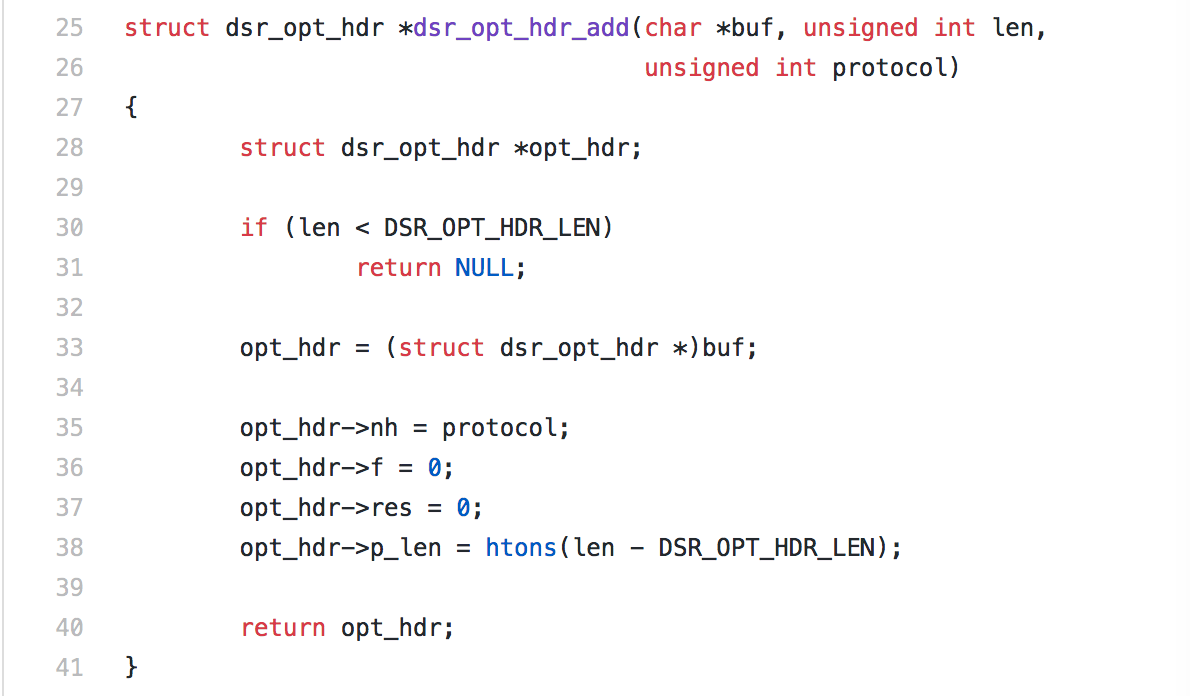
图7 dsr\_node UML结构图

一个dsr节点包括slave节点的设备信息slave\_dev，网络信息slave\_ifname，网络设备信息indev，节点状态，节点网络地址struct in\_addr。

### 3.1.2 dsr\_opt结构

dsr\_opt\_hdr\_add函数如下截图所示：

dsr-opt.c



dsr-opt.c

图8 dsr\_opt\_hdr\_add方法

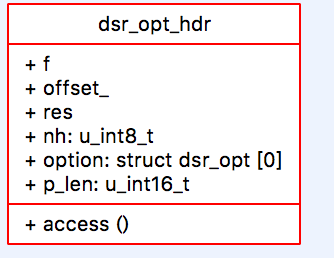


图8 dsr\_opt\_hdr结构

添加选项头的时候首先对选项头的长度进行判断，如果参数小于选项头的长度则返回空值。然后将buf指针强制类型转换为struct dsr\_opt\_hdr结构。这里用到了htons()将host类型转换为short number。

DSR的首部采用扩展性良好的TLV格式。除固定部分外，不同类型的选项(option)以TLV格式附加在固定部分之后。

处理dsr头部字段

如果是dsr\_src\_hdr :调用dsr\_src\_opt\_handler ，如果DSR\_IN\_OPT，调用dsr\_in\_src\_opt\_handler处理： 如果dsrhdr标志位中需要ack为1,则发送一个ack。如果DSR\_FORWARD\_OPT，调用dsr\_forward\_src\_opt\_handler。

1、重新设置接受地址为下一跳地址，源地质为本节点地址；

2、重新设置nexthop\_info\_entry中下一跳的ack请求时间，即：if\_in\_nexthop\_list判断转发的包的目的地址是否在下一跳得列表中。Get\_nexthop\_natime来获得请求下一个跳节点ack的时间。从nexthop\_info\_entry 中得到最近一次从下一收到的ack时间。改变Nexthop\_info\_entry中的标志位，需要ack ，计算下一次需要ack的时间；

 3、修改ip头目的地址为本节点地址，源地址为本节点地址。

#### 3.1.2.1 路由请求选项

1. rreq选项结构:

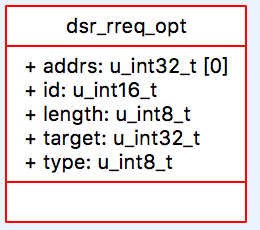
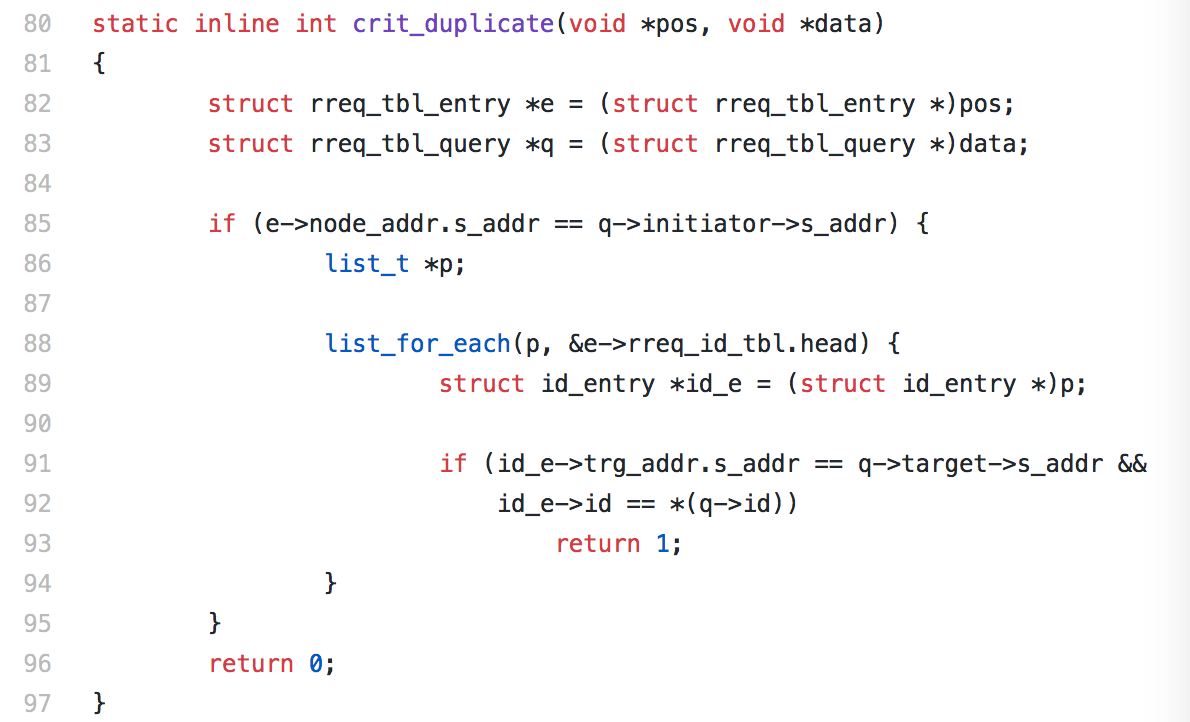


图9 dsr\_rreq\_opt UML图

dsr-rreq.c



dsr-rreq.c

图10 rreq的定义

使用内联函数，检查是否有重复的entry，使用list\_for\_each()遍历,如果检查到重复，则丢弃该重复消息。判断是否收到过这个请求，如果收到过，则返回，否则继续处理路由信息，添加从本节点到路由请求发起源的路由信息，然后针对网关修改，首先查找本地路由网关列表，没有则转发网关请求：iph->daddr = BCAST;如果本节点网关列表不为空，则还是make\_and\_send\_dsr\_reply来发送回复，将本地有的网关路由发送给请求节。

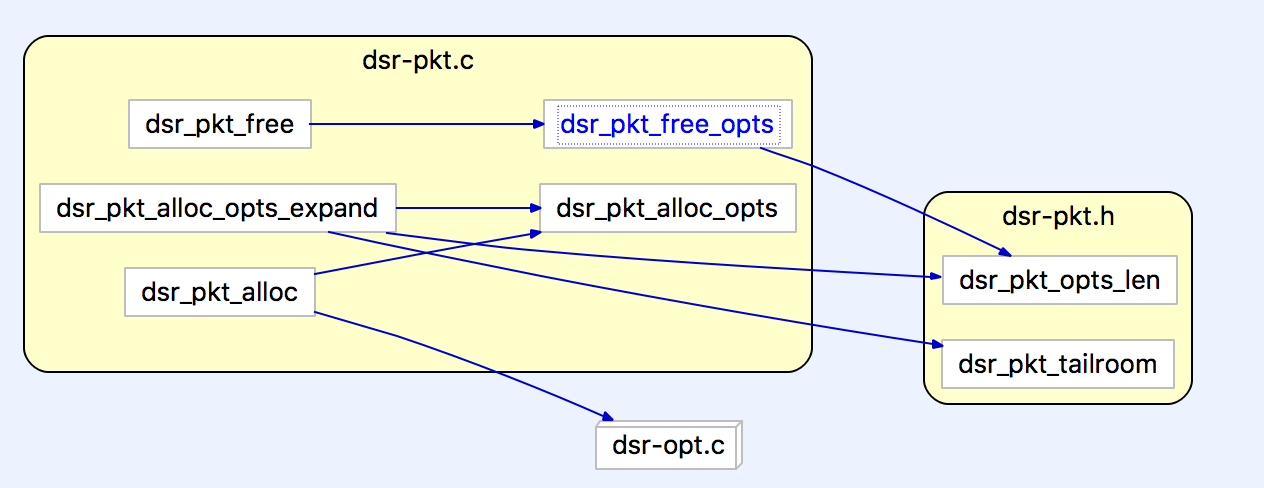


图11 dsr-pkt.c文件中的函数调用关系

2. 代码处理过程：

中间节点处理，中间节点维护<源节点地址、请求ID>序列对列表，重复RREQ检测；如果接收到的RREQ消息中的<源节点地址、请求ID>存在于本节点的序列对列表中；

如果接收到的RREQ消息中的路由纪录中包含本节点的地址；如果检测到重复，则中间节点丢弃该RREQ消息。

如果接收RREQ的节点是该路由请求的目的节点，则向发起RREQ的源节点返回RREP分组。将收到的RREQ分组的源节点地址、RREQ分组中携带的源路由节点地址列表和本节点的地址按顺序排列作为源路由封装在RREP(Route Reply)分组中发送给源节点，并将处理后的RREQ分组删除。

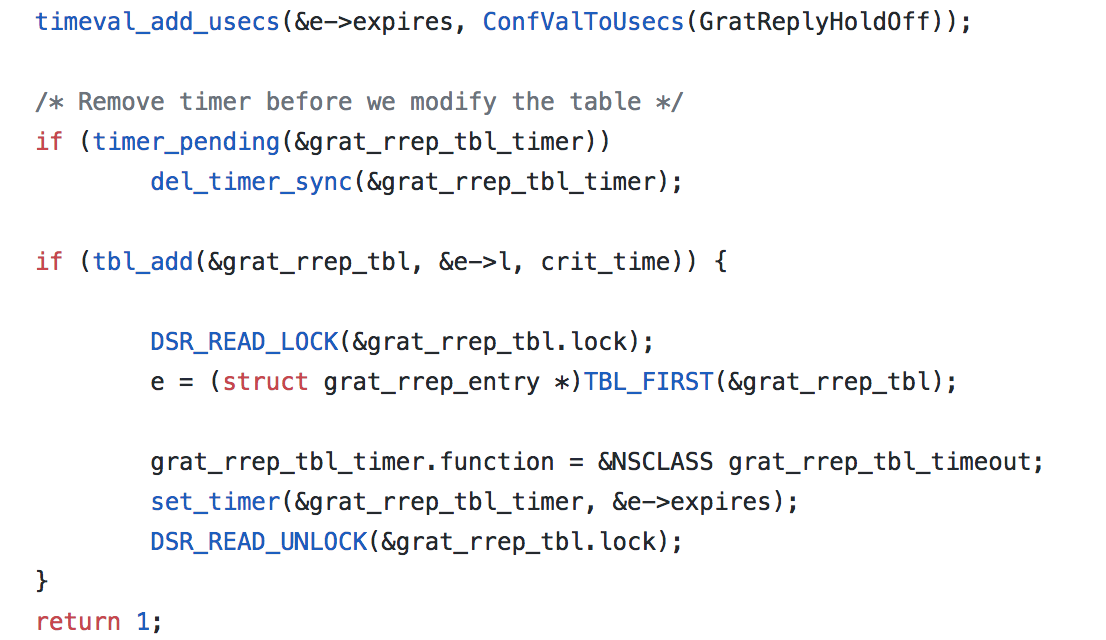
收到RREQ的节点检查自己是否已经包含在RREQ携带的源路由节点列表中，如果是则将RREQ分组丢弃。

如果是DSR\_REQ\_OPT ,调用dsr\_req\_opt\_handler如果是DSR\_IN\_OPT ，则调用dsr\_in\_req\_opt\_handler。

#### 3.1.2.2 路由应答选项

1. rrep选项结构:

dsr-rrep.c



dsr-rrep.c

图12 增加rrep表项函数

用于增加grat rrep table表项，新建两个结构体，一个用于查询，一个作为表项。在修改table之前移出timer。增加表项的时候，异步的增加timer,使用了timer\_pending()函数以及del\_timer\_sync()函数。在增加表项的时候现将dsr-read使用dsr-read-lock加锁，然后新建一个rrep\_tbl\_entry，重新设置Set\_timer，最后解锁DSR\_READ\_UNLOCK().

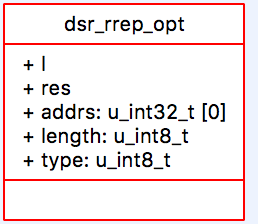


图13 dsr\_rrep\_opt结构 UML图

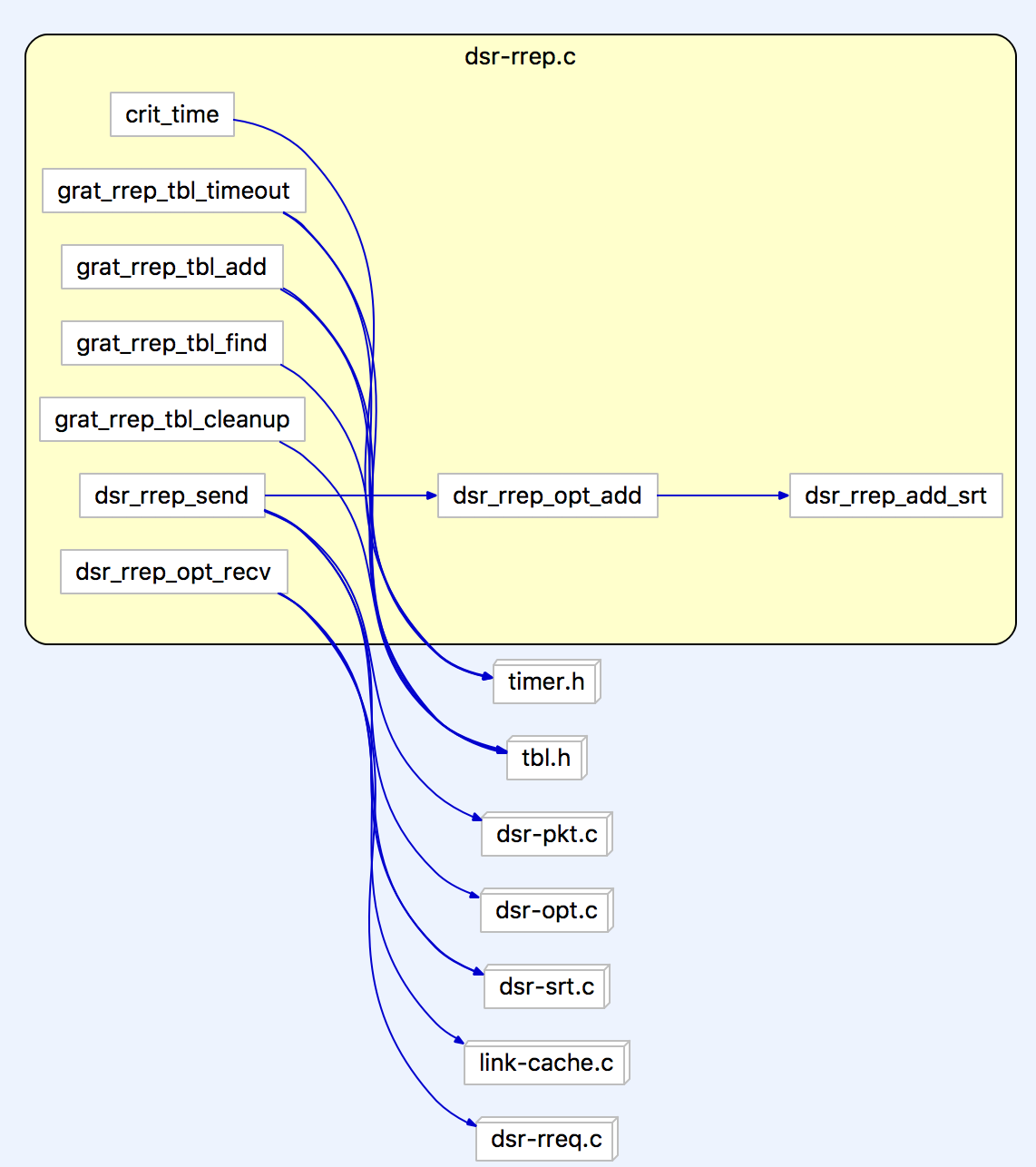


图14 dsr\_rrep.c文件中函数调用关系

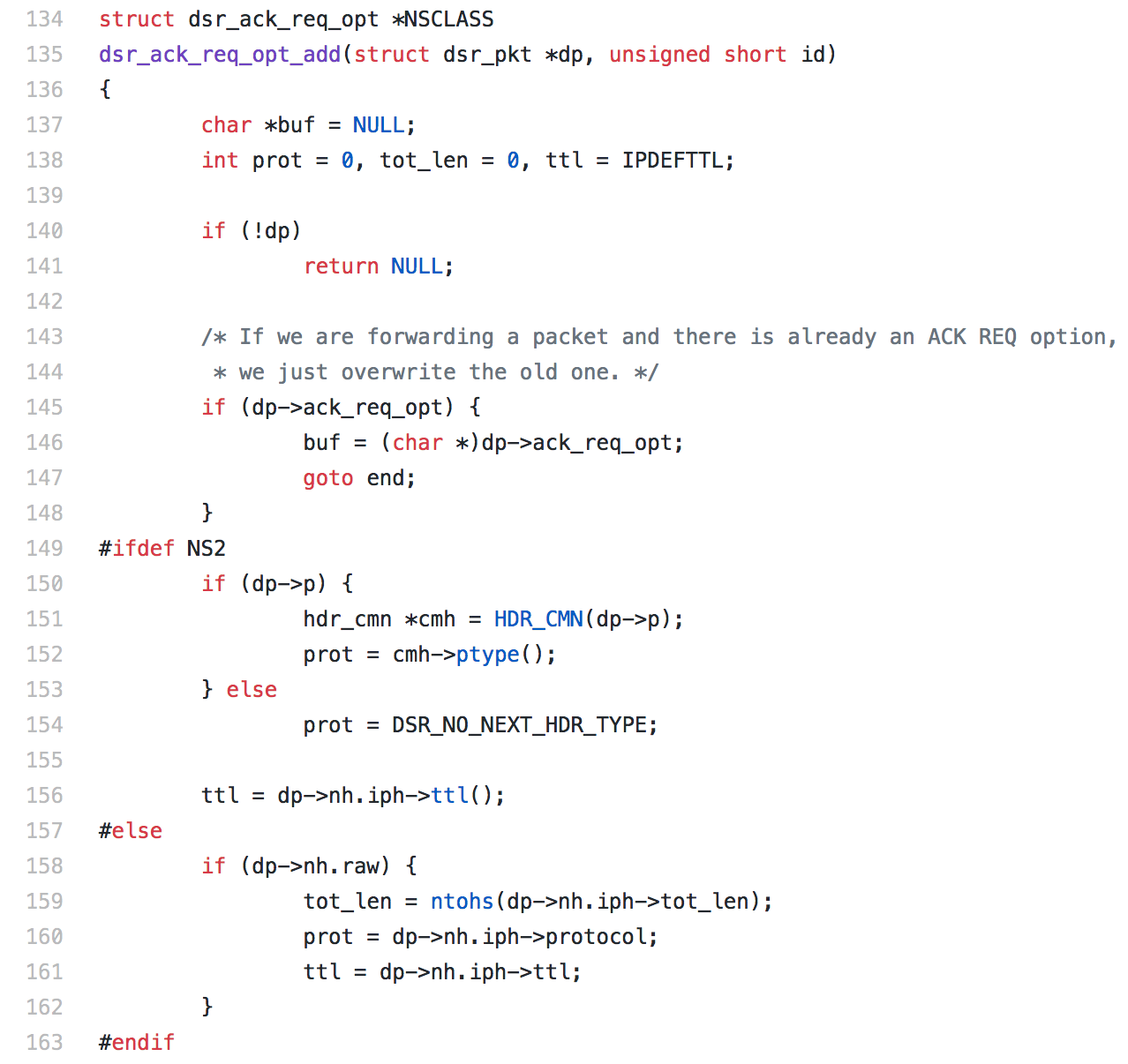
2. 处理过程：

中间节点回复RREP分组： 如果接收RREQ的中间节点在自己的路由缓存中找到通往RREQ分组目的节点的路径则要向源节点回复RREP分组.

#### 3.1.2.3 ACK请求选项

1. ACK请求选项结构:

dsr-ack.c



dsr-ack.c

图15 dsr\_ack\_req\_opt\_add方法

135~147：方法用于发送ack请求，首先构造packet,以及ack请求选项，如果没有成功生成ip\_header，抛出错误goto out\_err，构造dsr选项头，增加ack选项dsr\_ack\_opt\_add(buf, len, dp->src, dp->dst, id);

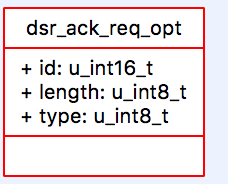


图16 dsr\_ack\_req\_opt结构

2. 处理过程：

dsr-ack.h ack请求选项以及ack选项的构成

dsr-ack.c 添加ack，发送ack，产生请求，发送请求，接收请求，接收ack。

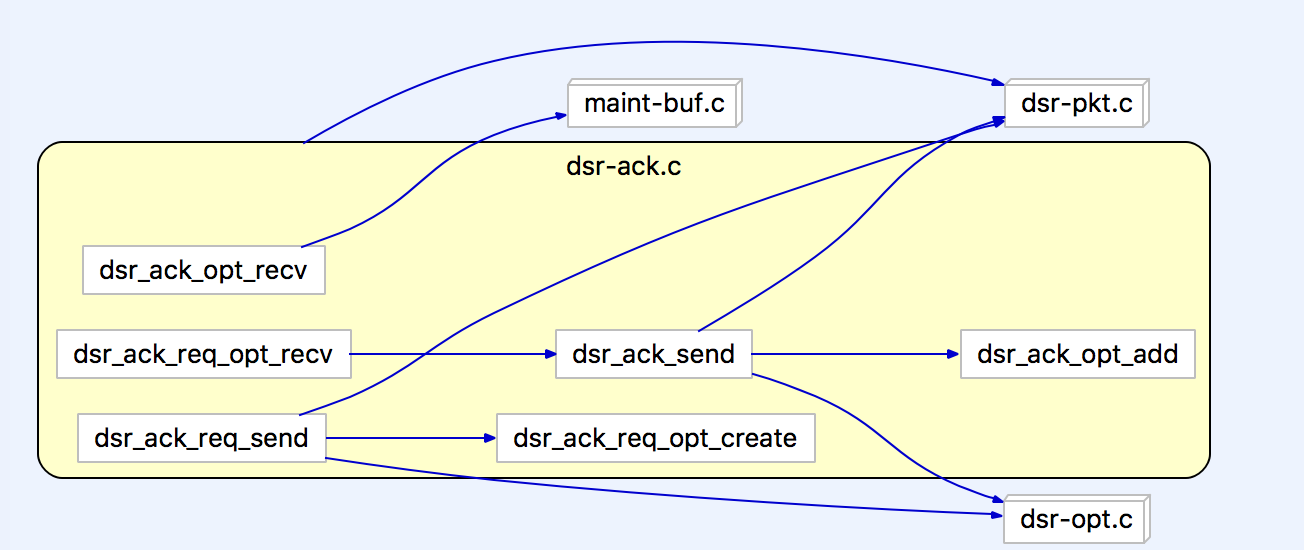


图17 dsr\_ack.c函数调用关系

当接收到dsr ACK请求选项时dsr\_ack\_req\_opt\_recv()调用dsr\_ack\_send()函数进行发送。

同时添加dsr ack选项，请求发送函数调用处理生成一个请求选项，然后调用dsr-pkt.c文件生成dsr packet, 调用dsr-opt.c以及maint-buf.c来接收dsr ACK选项并且存入缓存中。

dsr\_ack\_send方法用于发送ack请求，寻找源路由结点，如果没有找到则生成错误信息。构造packet长度，等于dsr选项头长度加上ack选项头长度。如果接收到错误信息，清理dsr packet。

#### 3.1.2.4 ACK选项

1. ACK选项结构:

 dsr-ack.c

dsr-ack.c

图18 dsr\_ack\_send方法

dsr\_ack\_send方法用于发送ack请求，寻找源路由结点，如果没有找到则生成错误信息。构造packet长度，等于dsr选项头长度加上ack选项头长度。如果接收到错误信息，清理dsr packet。

49~54：首先通过调用dsr\_rtc\_find()函数寻找到源路由节点，如果不存在源路由节点就输出错误日志 Debug(“No source route ”)，并返回-1；

56：packet的长度等于dsr选项头的长度加上dsr\_ack头的长度

58~65：构造dsr packet，将需要发送的包的地址赋值给packet，下一跳地址设为目的地址，packet的源路由地址设为my\_addr();

67~69: 错误处理部分，如果出错就将packet free，并返回-1；

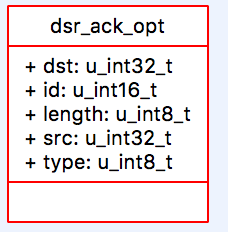


图19 dsr\_ack\_opt结构

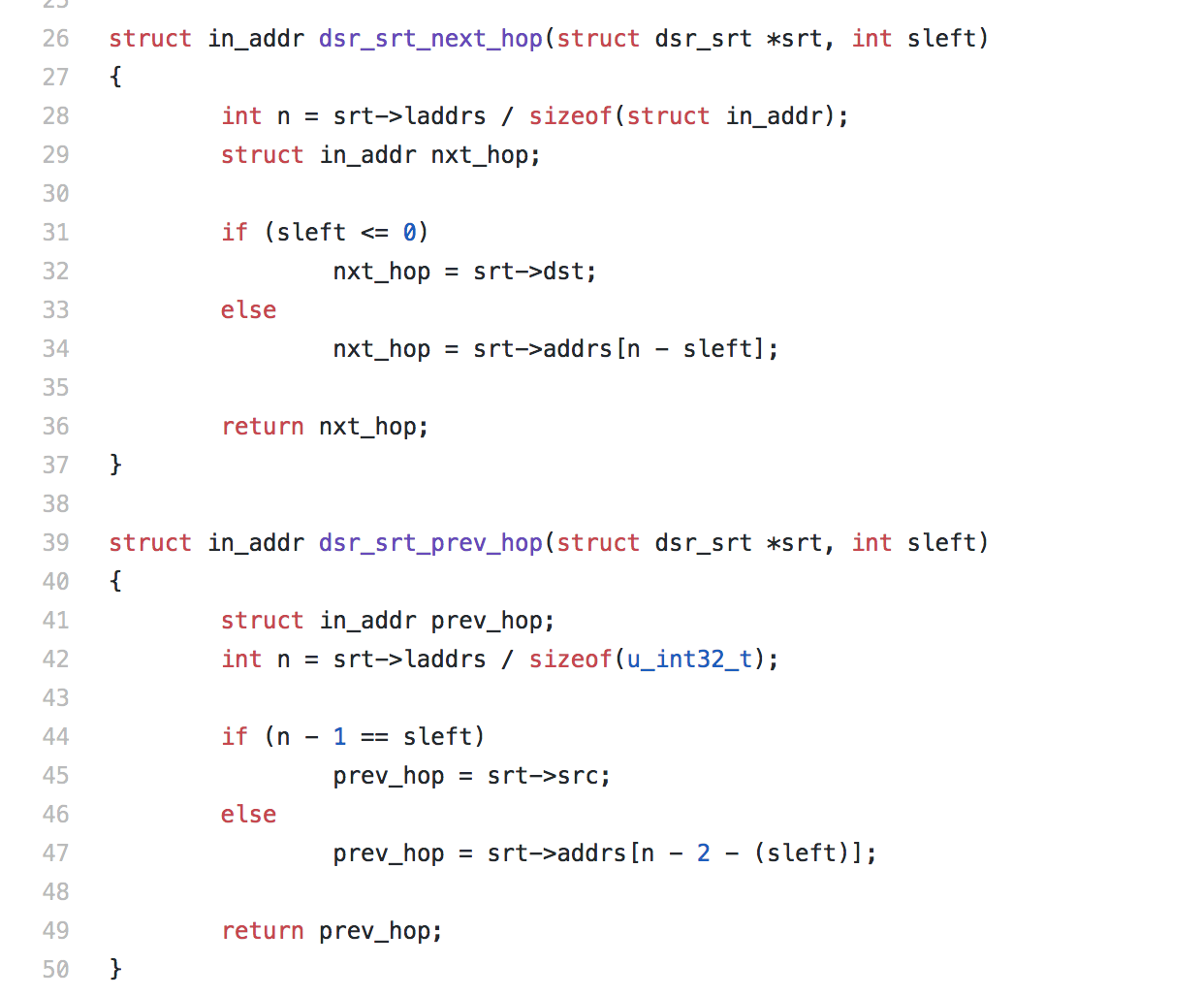
2. 作用：

dsr-ack.h ack请求选项以及ack选项的构成

dsr-ack.c 添加ack，发送ack，产生请求，发送请求，接收请求，接收ack。

#### 3.1.2.5 源路由选项

1. 源路由选项结构:

 dsr-srt.c

dsr-srt.c

图20 dsr\_srt\_next\_hop 与dsr\_srt\_prev\_hop 方法

dsr\_srt\_next\_hop实现下一跳的功能。dsr\_srt\_prev\_hop目的地址可以从srt→laddrs中取到，但保留in\_addr\_nxt\_hop可以大大加速转发速度。根据跳数hops最少来确定最佳路由。

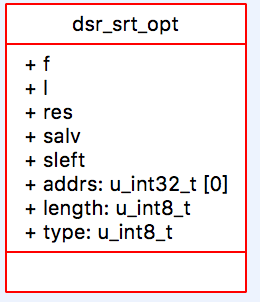


图21 dsr\_srt\_opt结构

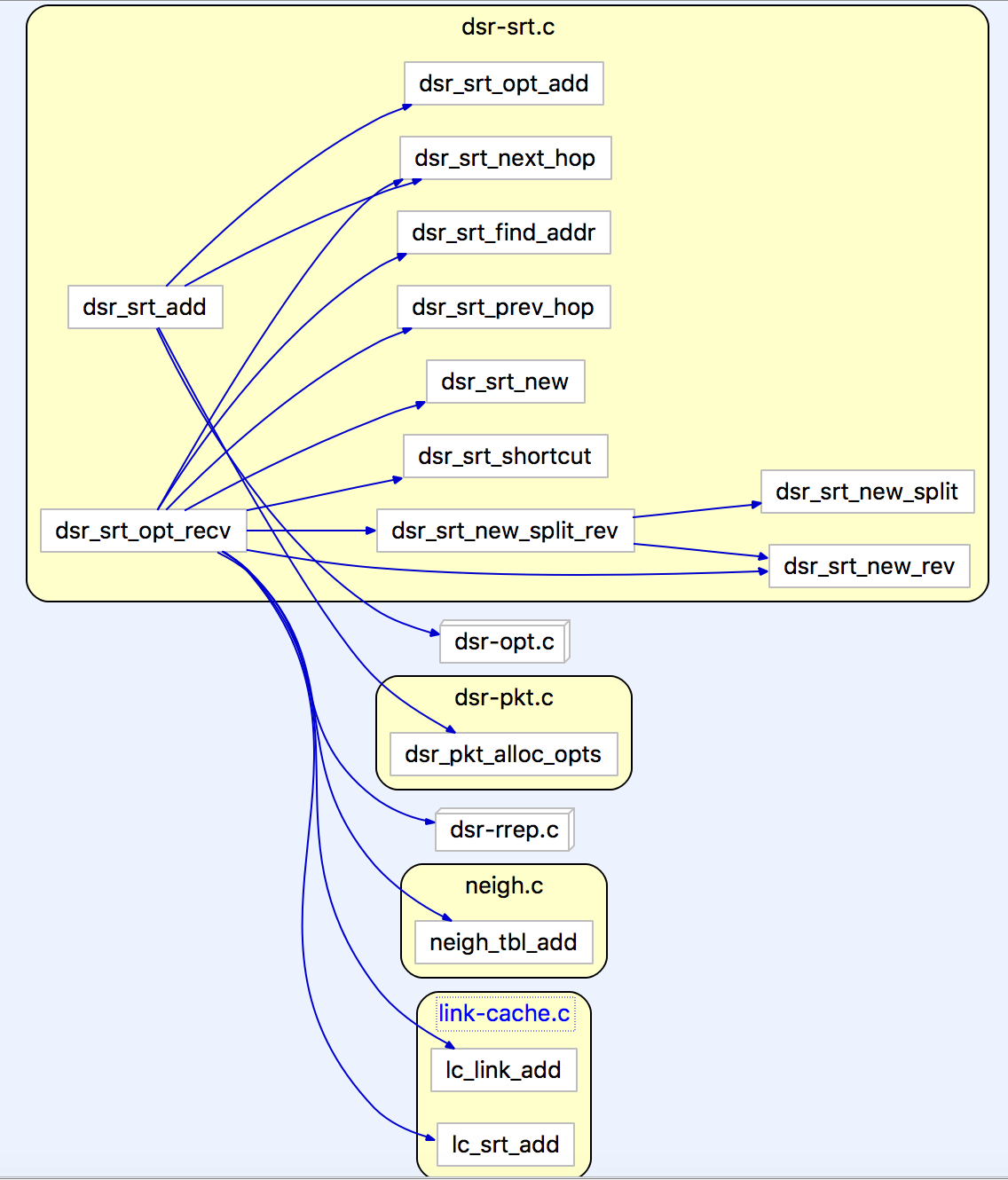


图22 dsr\_srt.c内部函数调用关系

2. 介绍

发送节点在分组中携带到达目的节点的路由信息（转发分组的完整的节点序列）– 不需要中间节点维护路由信息。节点缓存到目的节点的多条路由– 避免了在每次路由中断时都需要进行路由发现，因此能够对拓扑变化作出更快的反应。

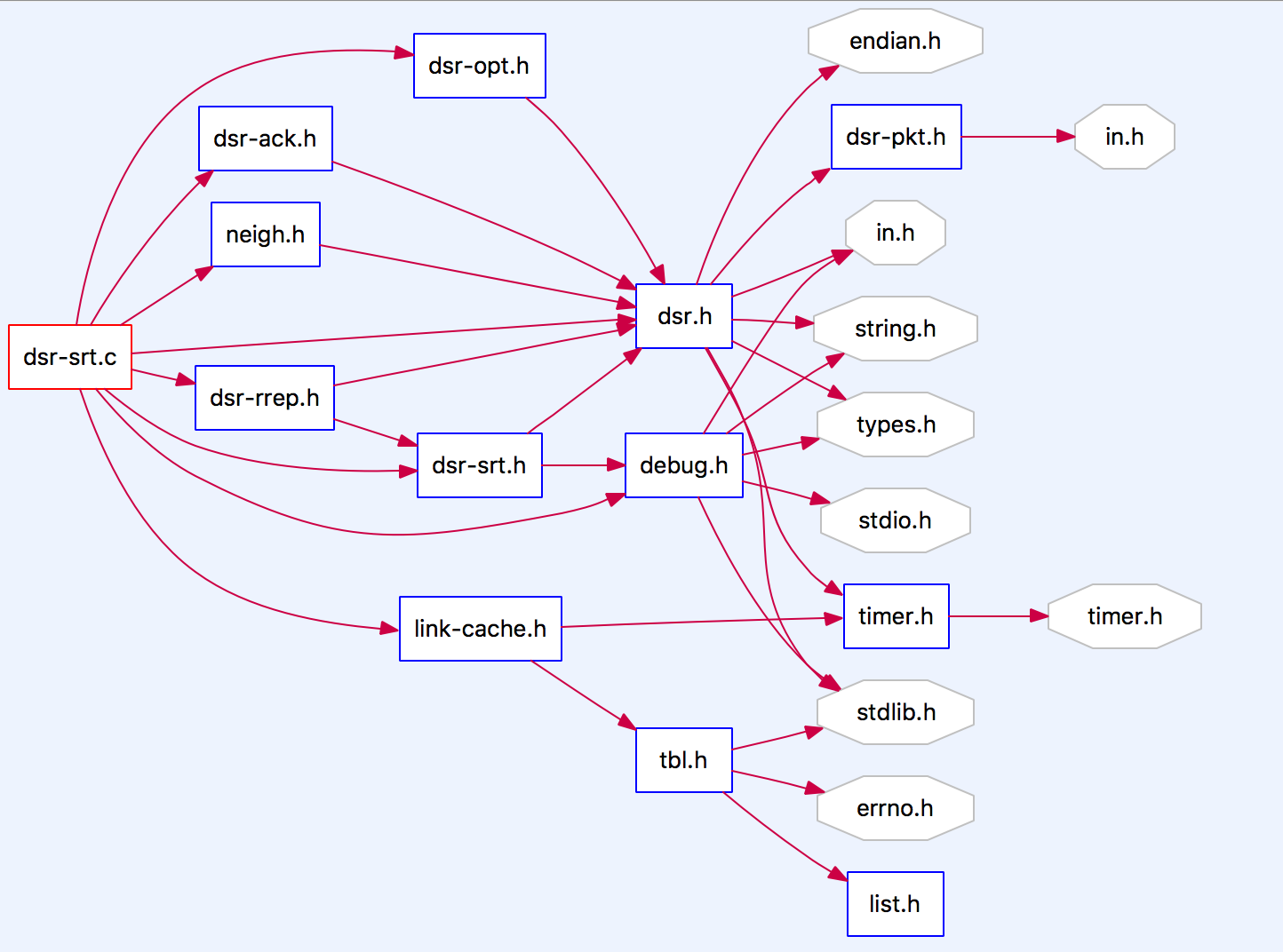


图23 dsr.h与各个文件之间调用关系

首先由dsr\_srt处理， dsr-srt.c主要负责构造源路由节点，通过调用dsr-ack.h处理正确应答以及ACK请求报文，通过neigh.h来处理邻居节点的信息，通过dsr-rrep.h来处理请求报文段的发送以及处理信息，通过link-cache.h来处理与链路层上的设备的交互。

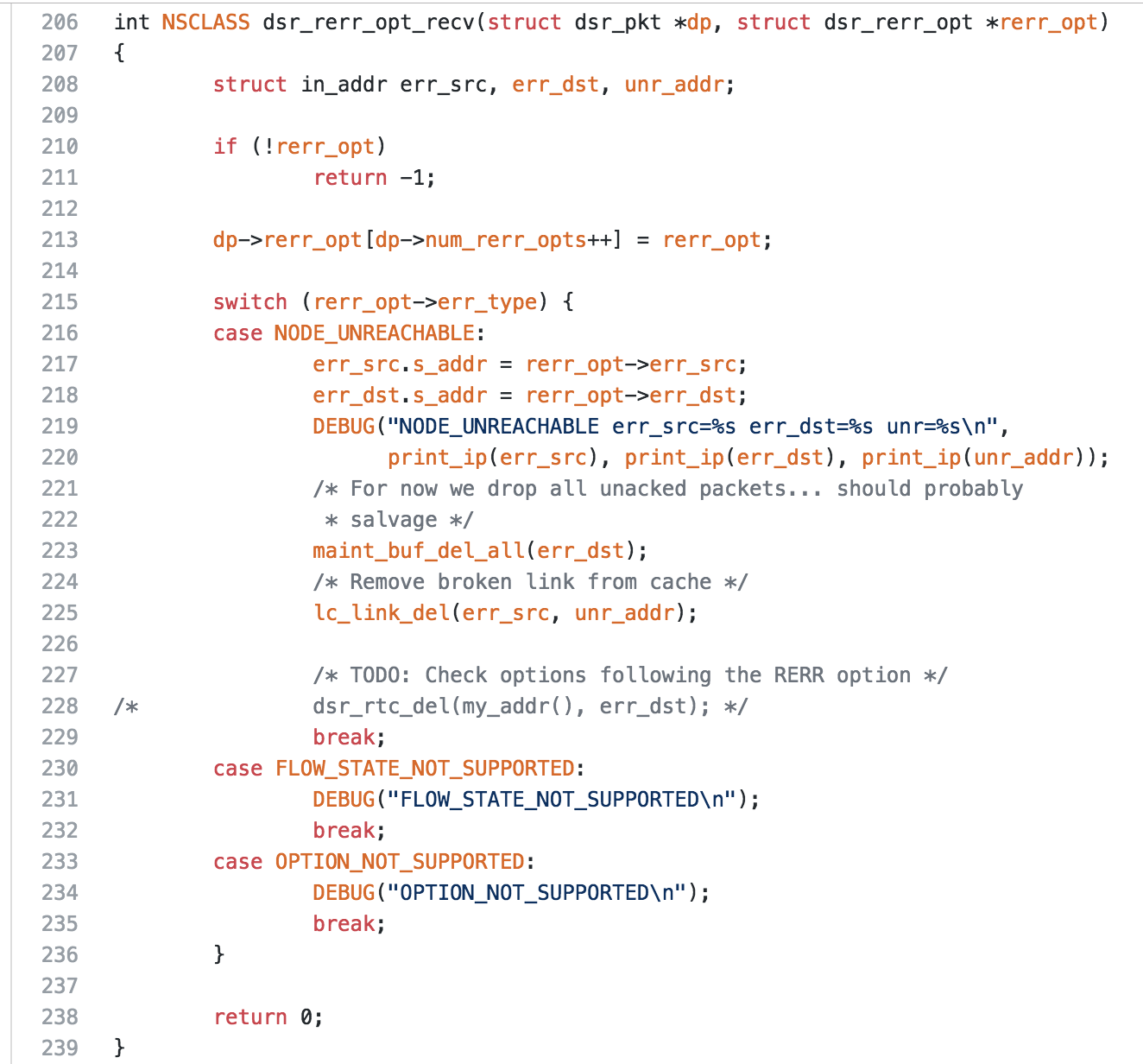
除此之外，dsr.h还使用了几个系统文件来帮助实现源路由部分的功能，分别是：endian.h提供网络字节顺序大小端，string.h, types.h, timer.h, errno.h通过一系列以E开头的错误信息选项来标识当前的出错信息。

dsr.c文件中提供了大多数的源路由功能，包括源路由选项的增加，上一跳，下一跳，查找源路由地址，源路由的接收。

使用了dsr-pkt.c文件中的dsr\_pkt\_alloc\_opts()函数来显式的构造dsr packet选项结构，使用了neigh.c文件中的neigh\_tbl\_add()来增加邻居节点的表项，使用了link-cache.c文件中的lc\_srt\_add()以及lc\_link\_add()函数来增加srt以及link选项。

## 3.2 rerr选项结构

1. rerr选项结构:

 dsr-rerr.c

dsr-rerr.c

图24 dsr\_rerr\_opt\_recv方法

接收错误选项，如果没有正确到达，则将该选项发给他的源节点及目的节点，删掉所有没有接收ack的包，将破损的链从缓存中移出。

如果在找到目的地之前减至０，则这个请求包将被丢弃。运用这个机制来发送一个非广播式路由请求，用以确定目标是否是 起始节点的邻居或临近节点是否存储有到达目的节点的路径。否则输入出信息OPTION\_NOT\_SUPPORTED

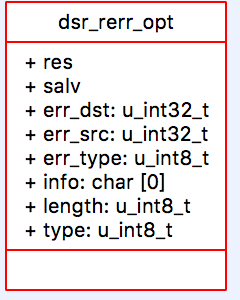


图25 dsr\_rerr\_opt结构

2. 代码介绍

网络拓扑的变化使得缓存的路由失效，影响和感染其它节点，使用该路由缓存的路由将不可用。当节点根据路由缓存回应RREP时，其它监听到此RREP的节点会更改自己缓存的路由，从而感染错误路由缓存，设置缓存路由的有效期，过期即删除。

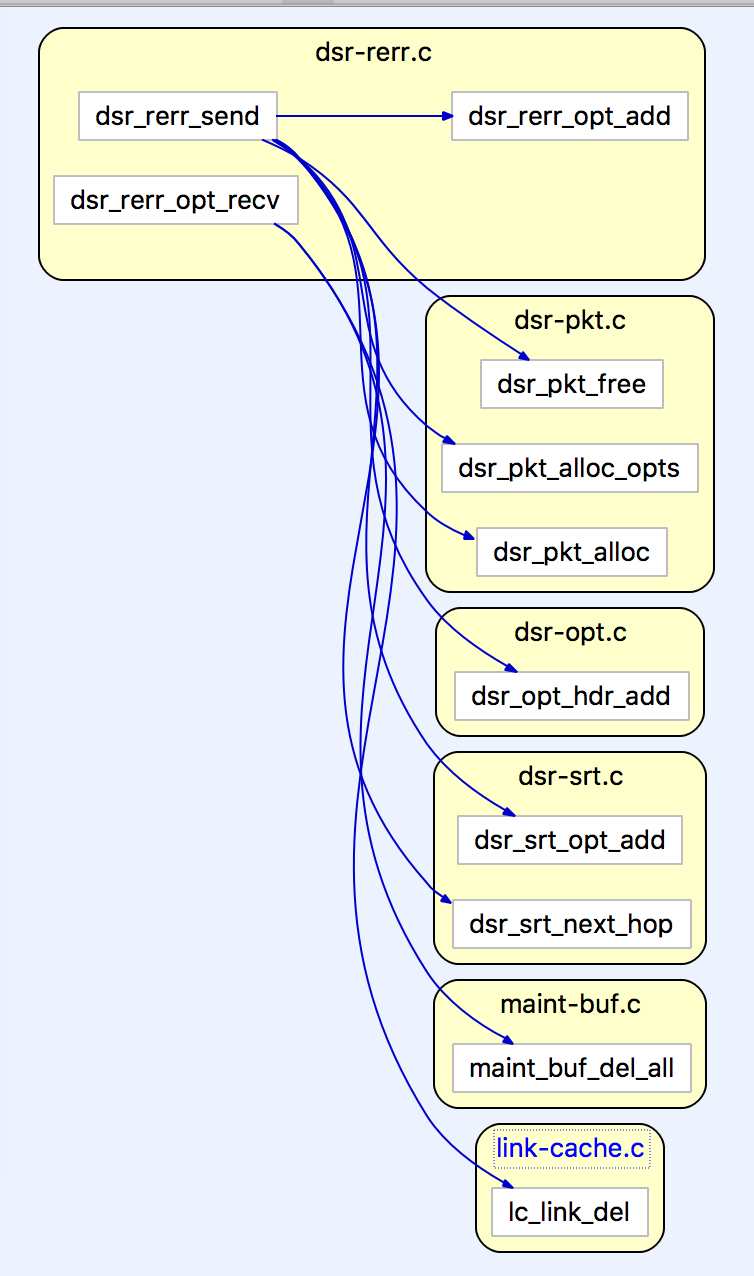


图26 dsr-rerr.c与各文件之间调用关系

## 3.3 maint\_buf选项结构

1. 路由维护缓存选项结构:

dsr-maint\_buf.c



dsr-maint\_buf.c

图27 dsr-maint\_buf.c函数部分截图

2. 代码介绍

515~525：如果数据分组被重发了最大次数仍然没有收到下一跳的确认，则节点向源端发送路由错误（Route Error）消息，并且指明中断的链路源端将该路由从路由缓存中删除。

527~532：如果源端路由缓存中存在另一条到目的节点的路由则使用该路由重发分组，否则重新开始路由发现过程。

536~541：如果packet的标志不为1或这不是一个ACK请求报文，则通过调用debug.c的调试日志输出信息Debug()

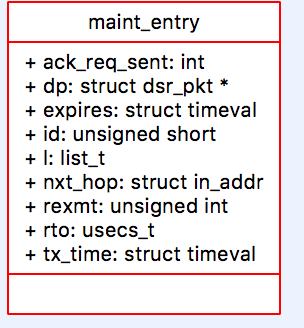


图28 maint\_entry结构

# 4 DSR协议基本原理

## 4.1 路由发现

路由发现的作用是使每个节点都能动态地找到到达其他节点的路径，当某个节点接收到一个路由请求包时，对请求包处理方法有以下４类：

1. 本次路由请求标识与起始端地址已经存在于最近可见的请求中，为防止重复路由请求包，放弃该请求包．
2. 节点地址已经存在于请求的路由记录中，也丢弃路由请求包以防产生循环路由．
3. 请求包的目标标识与某个节点的地址相吻合，则信息包中的全程路由记录得以确认，返回响应包到起始端．
4. 请求标识与节点地址不吻合，追加此节点地址到路由请求包的路由记录中，并重广播此请求．

当节点想要将数据发送到目标节点时，它首先在路由缓冲区中查询到目标节点的路由。 如果有，按此路由发送; 如果不是，则源节点启动路由发现过程。 路由发现过程使用洪泛路由。

当节点S需要向节点D发送数据, 但不知到节点D的路由，于是节点S就开始路由发现过程。源节点S洪泛“路由请求”分组Route Request (RREQ)，每个请求分组通过序列号和源节点S标识唯一确定。

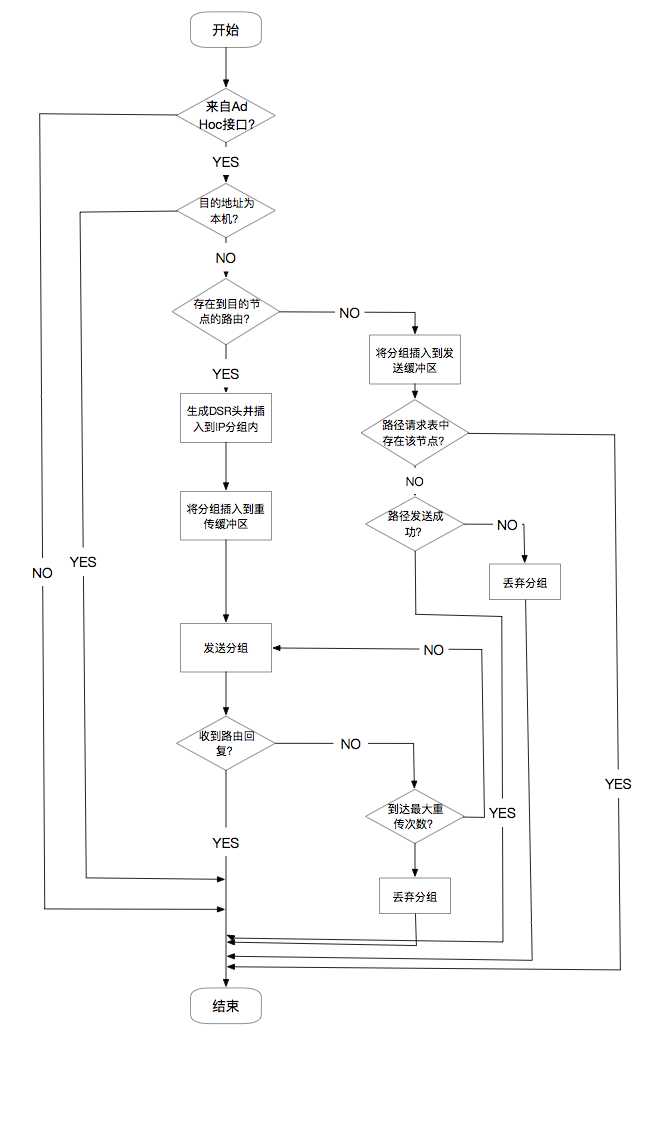


图29 DSR路由发现流程图

### 4.1.1路由请求

源节点向邻居节点广播路由请求（RREQ：Route Request）目的节点地址

路由记录：纪录从源节点到目的节点路由中的中间节点请求ID

中间节点接收到RREQ后，将自己的地址附在路由纪录中。

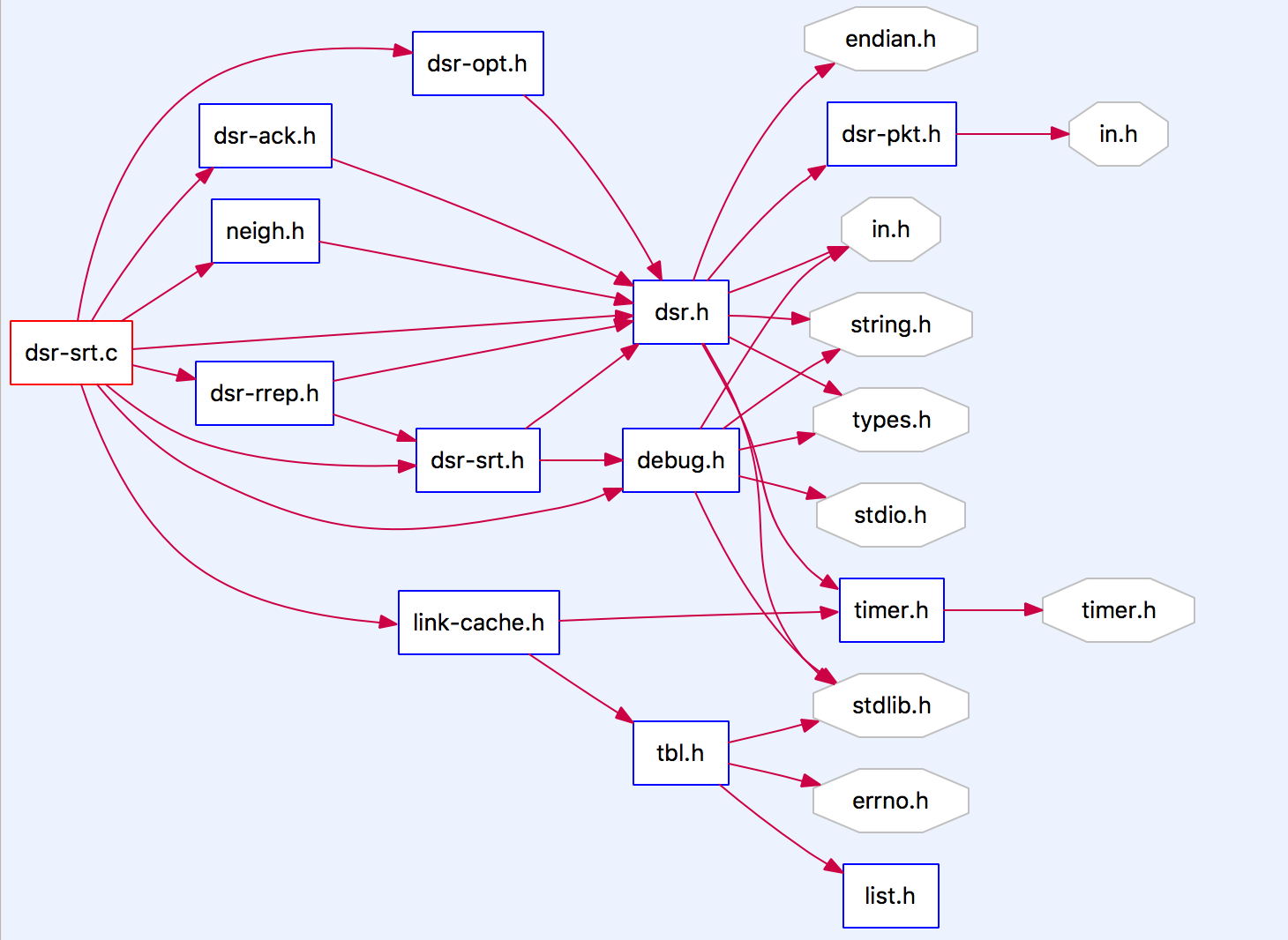


图30 路由请求调用关系

### 4.1.2 重复RREQ检测

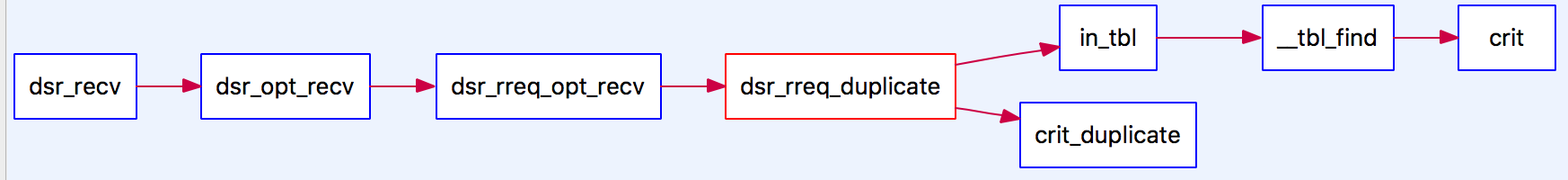


图31 重复RREQ检测函数关系

如果接收到的RREQ消息中的<源节点地址、请求ID>存在于本节点的序列对列表中

如果接收到的RREQ消息中的路由纪录中包含本节点的地址

如果检测到重复，则中间节点丢弃该RREQ消息

函数处理过程是，首先接收dsr数据报，接收到dst路由选项，接收dsr路由请求选项，检查是否有重复的rreq，使用函数dsr\_rreq\_duplicate()进行检测，这个函数中使用到了in\_tbl()函数以及crit\_duplicate()函数。

## 4.2 路由维护



图32 DSR路由应答流程图

调用了make\_and\_send\_dsr\_data\_packet进行处理 具体过程：

如果hoplist是空 则调用make\_dsr\_source\_route\_option ，在这个函数中，首先比较目的地址是否为本网段，如果不是本网段则需要查找本地网关列表

如果没有网关信息，则发起网关发现：gateway\_discovery()

否则检查到网关/内网地址由，如果没有路由，则发起路由请求，将发送的数据暂时保存在发送缓冲区中。

总结一下make\_dsr\_sour\_route\_option 的功能，就是检查网关信息和路由信息，没有相关信息则发起路由请求，把要发送的数据存到发送缓冲区中。

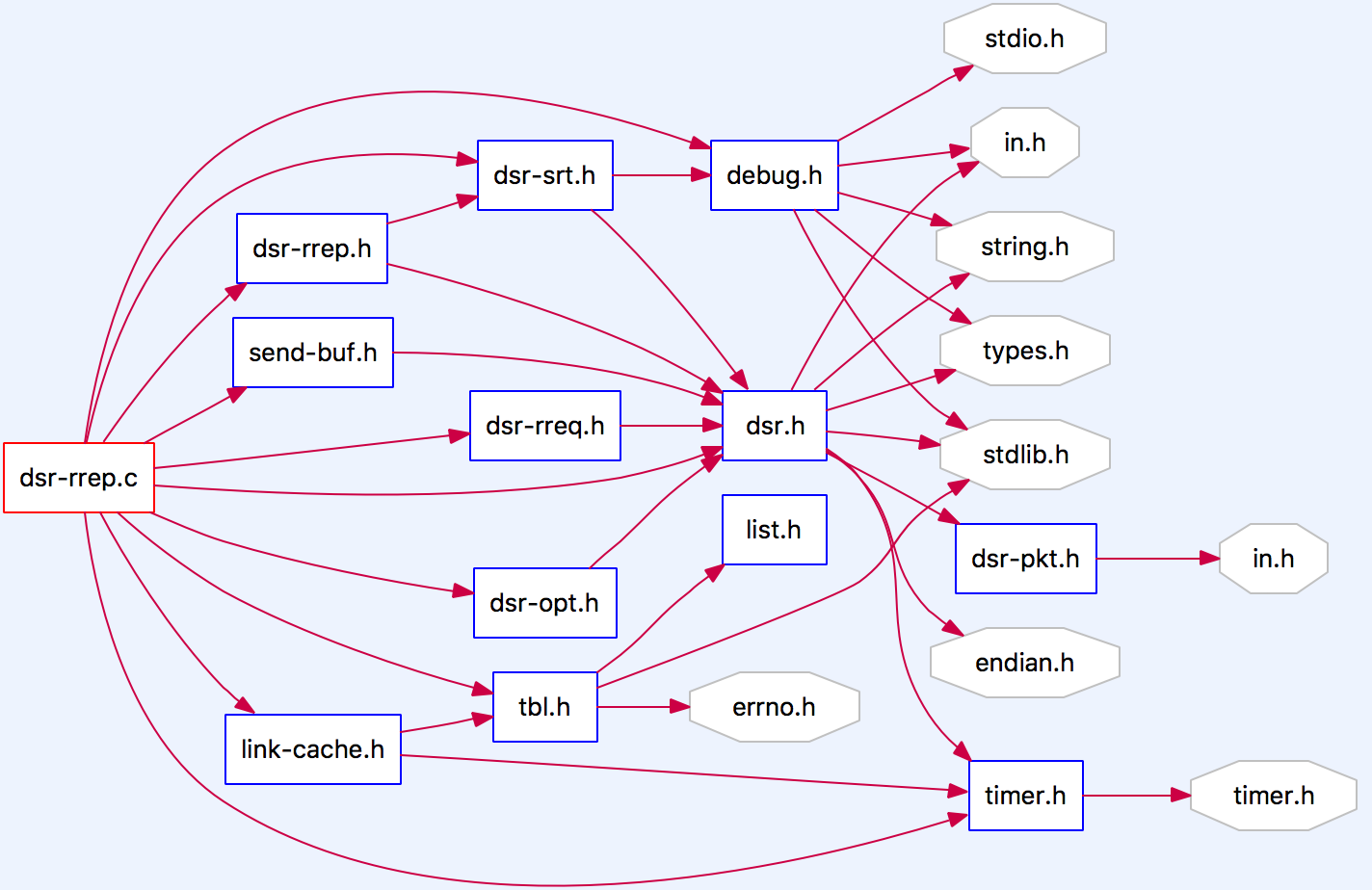


图33 函数调用关系

目的节点收到RREQ后，给源节点返回路由应答（RREPRoute Reply）消息，拷贝RREQ消息中的路由纪录，源节点收到RREP后在本地路由缓存中缓存路由信息。此处dsr-rrep.c文件调用link-cache.h文件来新增lc路由选项以及路由表，通过调用send-buf.h文件在本地路由缓存中缓存路由信息，通过调用debug.h文件输出错误处理调试日志。

## 4.3 缓存机制

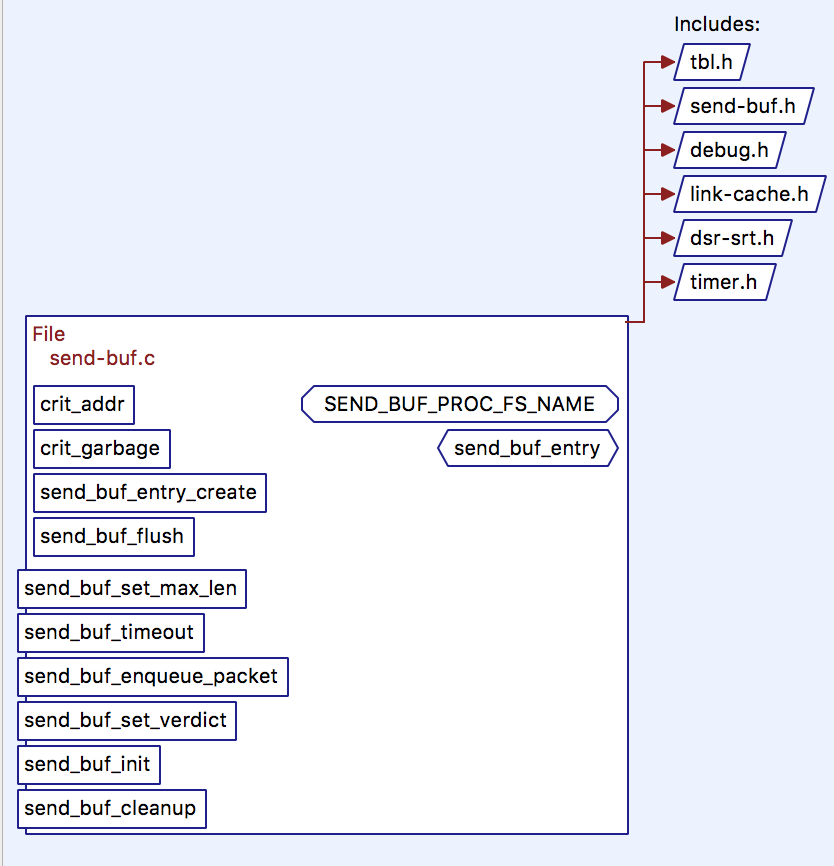
每个路由请求报文都含有一个多跳限制，用于限制传送路由请求的中介节点的数目．随着请求的传送，这个限制逐渐减小，如果在找到目的地之前减至０，则这个请求包将被丢弃，运用这个机制来发送一个非广播式路由请求，用以确定目标是否是起始节点的邻居或临近节点是否存储有到达目的节点的路径（有效地将邻节点的缓存器作为起始节点缓存器的延伸）。

如果没有接收到路由回复，则发送广播式路由请求。

或者采用扩展环方式来寻找目的节点，渐渐地探测到目标而不需要将路由请求在 整个网络中广播。

源路由的路由机制，在每一个分组的头部都携带整条路由的信息，路由器按照该路由纪录来转发分组。这种机制最初被IEEE802.5协议用在由桥互连的多个令牌环网中寻找路由。

### 4.3.1 Send-buf TBL



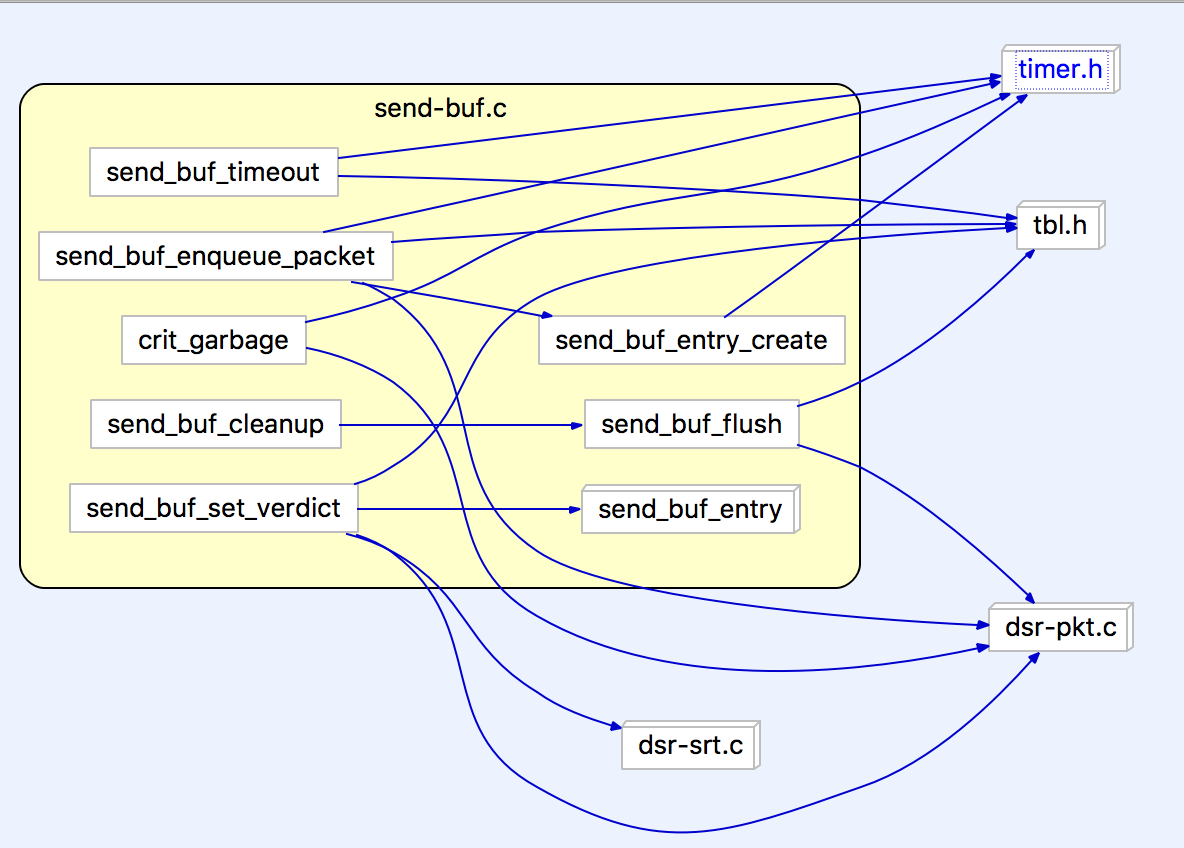
 图34 Declaration: send-buf

图35 send-buf.c内部函数调用关系

当节点要传送数据分组时，源节点先检查缓存中是否有到信宿的路由信息，若有非过期的路由则可直接采用，否则泛洪广播发送路由请求分组。

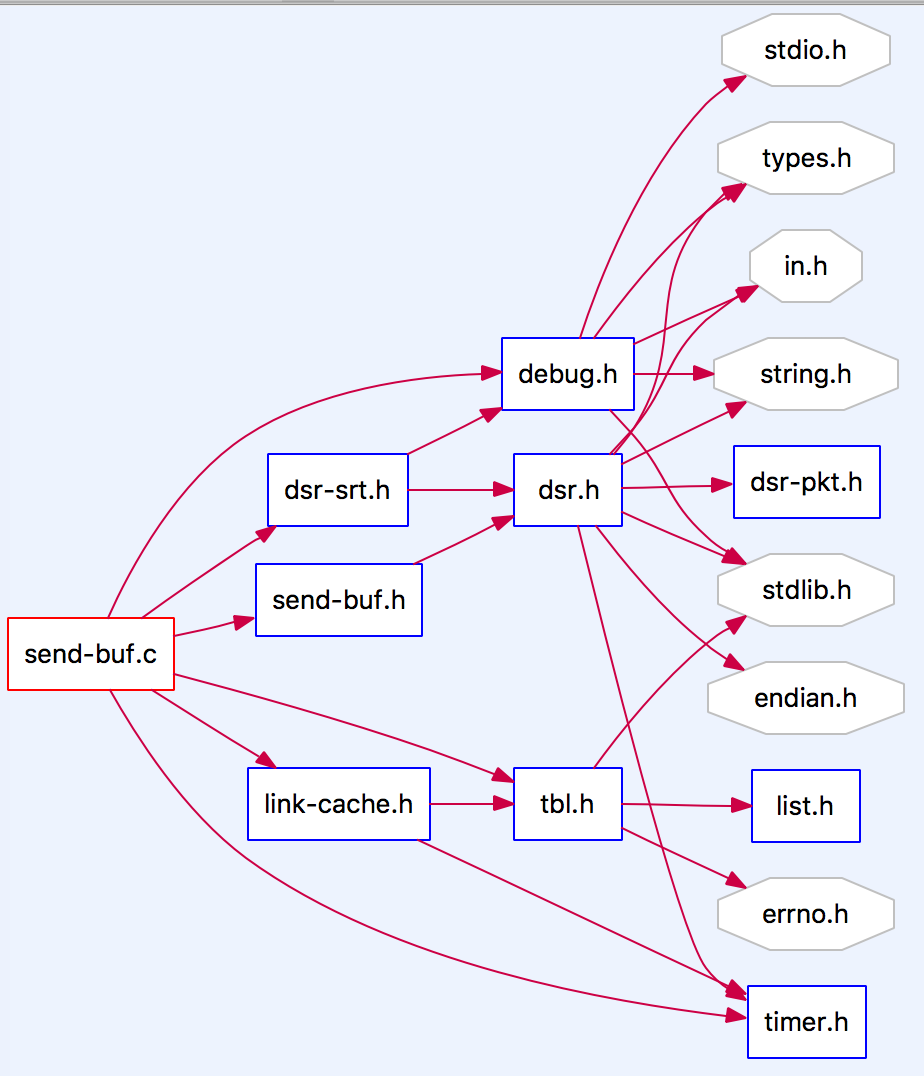


图36 send-buf.c与其他文件调用关系

### ../../../屏幕快照%202018-12-18%20上午10.23.27.png4.3.2 Maint-buf TBL

图37 maint-buf.c内部结构

在此过程中需要对已建立的路由进行维护。源节点通过路由维护机制可以检测出网络拓扑的改变,从而知道到目的节点的路由是否可用。当路由维护探测到某条使用中的路由出现了问题时,就会发送RERR(路由错误报文)给源节点，源节点在收到该 RERR后,就会从它的路由缓存中删除所有包含该故障链路的路由，并重新发起一个路由发现过程。

## 4.4 应答机制

### 4.4.1 ack确认处理机制

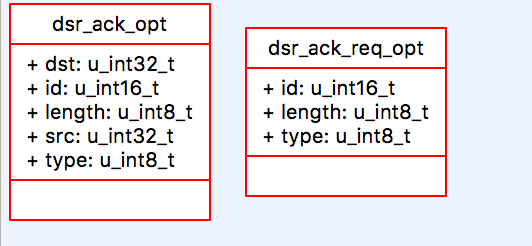


图38 ack\_opt 与 ack\_req\_opt 结构

dsr-ack.h ack请求选项以及ack选项的构成

dsr-ack.c 添加ack，发送ack，产生请求，发送请求，接收请求，接收ack

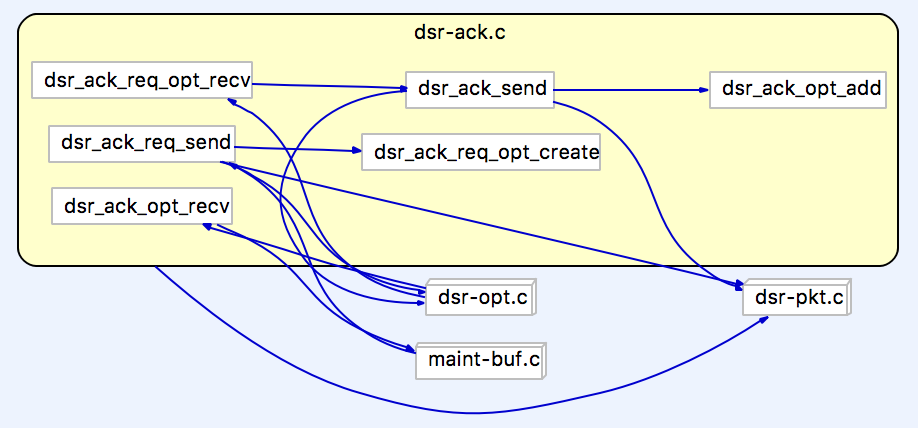


图 39 dsr-ack.c文件中函数关系以及其他文件相互关系

当接收到dsr ACK请求选项时dsr\_ack\_req\_opt\_recv()调用dsr\_ack\_send()函数进行发送，同时添加dsr ack选项，请求发送函数调用处理生成一个请求选项，然后调用dsr-pkt.c文件生成dsr packet, 调用dsr-opt.c以及maint-buf.c来接收dsr ACK选项并且存入缓存中。

### ../../../屏幕快照%202018-12-18%20上午10.34.46.png4.4.2 rrep 应答处理机制

图40 Declaration: rrep

路由维护是运用数据层，点对点的确认来提供丢失的或被破坏的信息包的提早发现和重传输．当 运用源路由发送信息包时，每个传送节点都是依次确认。

目的节点收到RREQ后，给源节点返回路由应答（RREP）消息，拷贝RREQ消息中的路由记录。源节点收到RREP后在本地路由缓存中缓存路由信息。

DSR路由应答（链路为双向的）：

（1）当目的节点D一接到RREQ分组，就发送RREP分组

（2）RREP分组中包含有RREQ分组中从源节点S到目的节点的路由纪录（前向路）

（3）RREP分组按RREQ分组的路由纪录进行反向传送。

DSR路由应答（链路为单向的）：

此时，目的节点执行和源节点相同的路由发现过程，

所不同的的是，目的节点的RREQ分组捎带传送RREP分组。



图41 dsr-rtc-simple.c内部函数调用

## 4.5 错误处理

如果数据分组被重发了最大次数仍然没有收到下一跳的确认，则节点向源端发送路由错误（Route Error）消息，并且指明中断的链路，源端将该路由从路由缓存中删除，如果源端路由缓存中存在另一条到目的节点的路由则使用该路由重发分组，否则重新开始路由发现过程。

错误路由缓存，网络拓扑的变化使得缓存的路由失效，影响和感染其它节点，使用该路由缓存的路由将不可用，当节点根据路由缓存回应RREP时，其它监听到此RREP的节点会更改自己缓存的路由，从而感染错误路由缓存，设置缓存路由的有效期，过期即删除。

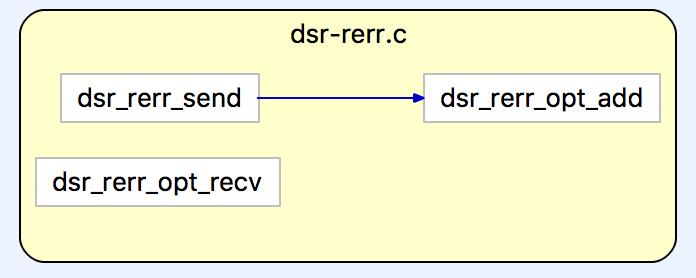


图42 错误处理函数

# 5 DSR协议特点

1）节点不需要周期性的发送路由广播分组，无需维护去往全网所有节点的路由信息，能自然而然的消除路由环路，而且能提供多条路由，可用于单向信道；

2）支持中间节点的应答，能使源节点快速获得路由，但会引起过时路由问题；

3）每个分组都需要携带完整的路由信息，造成开销较大，降低了网络带宽的利用率，不适合网络直径大的自组网，网络扩展性不强；

## 5.1 DSR协议优点

１）DSR不需要周期性地广播路由信息，从而降低了网络带宽开销，也节省了移动主机的电源能量（主机在不忙碌时可以通过将其进入“睡眠”或 “备用”模式而减少能源使用）。

２）传统路由协议在固定网中往往构造的是双向对称路由，在无线移动环境中，传播和节点周围的干扰情况不同，传输两端状况也有差异，采用双向对称路由会出现冗余“链接”，采用动态ＤＳＲ可以去除这些冗余。

３）传统路由协议无法适应网络拓扑结构动态变化，ＤＳＲ能够迅速适应节点移动的变化，协议的所有操作都是基于按需的，允许数据包动态的根据需要对当前路径的变化做出反应，但当这些变化没有发生时不需要任何路由协议开销。

## 5.2 协议缺点

1）随着路经跳数的增加，分组头长度线性增加、开销大。

2）路由请求分组RREQ采用洪泛发向全网扩散，导致网络负荷大。

3）来自邻居节点的RREQ分组在某个节点可能会发生碰撞，解决办法是：在发送RREQ分组时引入随机时延。

4）当源节点发送路由请求分组RREQ时，可能会收到多个节点缓存的到达目的节点的路由信息，引起竞争。

# 6 总结

动态源路由协议(Dynamic Source Routing Protocol，DSR)是由移动节点组成的多跳无线Ad Hoc网络中一种简单和行之有效的路由协议。协议允许任一结点动态发现到达Ad Hoc网络中其它任意节点的路由，所有的路由信息由DSR自动地进行维护。每个DSR头部都携带了到达目的节点的完整的路由跃点列表(hop list)，中间节点只需简单地对分组进行转发即可。同时DSR协议完全按需(on-demand)的特性可以显著减少路由协议的开销，减少了分组冲突的概率并减少了潜在的大规模的路径更新信息的传播。

DSR的特点在于使用了源路由的路由机制，在每一个分组的头部都携带整条路由的信息，路由器按照该路由纪录来转发分组。这种机制最初被IEEE802.5协议用在由桥互连的多个令牌环网中寻找路由。

DSR借鉴该机制，并结合了按需路由的思想。DSR协议使用源路由，采用Cache（缓冲器）存放路由信息，且中间节点不必存储转发分组所需的路由信息，网络开销较少，但存在陈旧路由。

通过这次大作业，我们学会了如何从零开始阅读源代码，我们先使用了老师推荐的阅读工具，理清了各个文件之间的关系和他们的作用，还了解了文件中变量的结构以及各个函数之间的调用关系。然后根据网上查阅的相关资料，更加进一步的深入了解了动态源路由协议(Dynamic Source Routing Protocol，DSR)，理解了动态源路由协议的整个处理机制以及操作流程。

通过阅读相关的期刊信息以及论文，我们了解到当前动态源路由协议大多实现在Linux系统下面，并且协议的实现都有相关的优化。

此后我们在了解清楚了动态源路由协议的整体框架以及协议流程之后，我们回归代码，开始对代码中的细节进行摸索，结合了高级C语言上学到的socket编程语法知识，更加清楚的明白了很多函数的处理办法以及整个过程中一些功能的实现。

综上，本次大作业的学习历程让我们了解到了学习一个路由协议乃至学习一部分新知识新技术的可行思路，也学习到了如何高效的去阅读源码，如何有条理的去从源码中剖析出框架思路，希望以后可以将本次学习到的东西更好的应用到学习中。