**网络协议栈分析与设计课程大作业**

|  |
| --- |
| **DSR路由协议代码分析** |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | 学号 | 姓名 | 班级 | 分工 | 成绩 | | 201692049 | 苏瑞 | 软网1602 | 1.代码分析  2.文档编写  3.排版 |  | | 201692458 | 陆宇韬 | 软网1602 | 1.代码分析  2.文献搜集  3.文档查改 |  | |

目录

[第一章 引言 3](#_Toc533762794)

[第二章 代码介绍 4](#_Toc533762795)

[2.1 符号与缩写 4](#_Toc533762796)

[2.2 文件介绍 4](#_Toc533762797)

[2.3 主要结构体定义 5](#_Toc533762798)

[第三章　DSR工作流程 6](#_Toc533762799)

[3.1系统流程图 6](#_Toc533762800)

[3.2 路由发现 7](#_Toc533762801)

[3.3 路由维护 9](#_Toc533762802)

[3.4 函数调用图 9](#_Toc533762803)

[第四章 DSR 路由协议的实现 10](#_Toc533762804)

[4.1 常规包处理 10](#_Toc533762805)

[4.1.1初始化一个包（分组） 10](#_Toc533762806)

[4.1.2将DSR选项头添加进包中 10](#_Toc533762807)

[4.1.3将DSR源路由选项添加进包中 11](#_Toc533762808)

[4.1.4处理一个接收到的包 12](#_Toc533762809)

[4.1.5处理收到的DSR源路由选项 14](#_Toc533762810)

[4.2 路由发现 17](#_Toc533762811)

[4.2.1一个例子 17](#_Toc533762812)

[4.2.2 发起路由请求 18](#_Toc533762813)

[4.2.3 处理接收到的路由请求选项 21](#_Toc533762814)

[4.2.4 使用路由缓存生成路由应答（“缓存路由应答”） 24](#_Toc533762815)

[4.2.5发起路由应答 25](#_Toc533762816)

[4.2.6防止路由应答风暴 27](#_Toc533762817)

[4.2.7 处理接收到的路由应答选项 27](#_Toc533762818)

[4.3 路由维护 28](#_Toc533762819)

[4.3.1 使用链路层ACK 28](#_Toc533762820)

[4.3.2 使用被动ACK 28](#_Toc533762821)

[4.3.3 使用网络层ACK 29](#_Toc533762822)

[4.3.4 发起路由错误 31](#_Toc533762823)

[4.3.5 处理接收到的路由错误选项 33](#_Toc533762824)

[4.3.6 补救一个包 34](#_Toc533762825)

[第五章 总结 36](#_Toc533762826)

# 引言

DSR协议由“路由发现”和“路由维护”两大机制组成。协议的所有方面都是按需操作的，允许DSR的路由包开销自动扩展到只需要对当前使用的路由中的更改做出反应即可，当节点开始移动或通信模式发生变化时，DSR的路由包开销自动扩展到只需要跟踪当前使用的路由。

DSR不使用任何周期性路由广告、链路状态感知或邻居检测包，也不依赖于网络中任何底层协议的这些功能。这种完全随需应变的行为和周期性活动的缺乏，使得DSR导致的开销包的数量可以一直伸缩。

为了响应单个路由发现(以及通过从偷听到的其他包路由信息)，节点可以学习并缓存到任一目的地的多条路由。这种对多个路由的支持使得对路由更改的反应更加迅速，因为具有多个路由到目的地的节点可以尝试另一个缓存路由(如果它一直使用的路由失败)。

该协议允许多个路由到任一目的地，并允许每个发送方选择和控制路由数据包时使用的路由，例如用于负载平衡或增强健壮性。DSR协议主要是为多达200个节点的移动自组织网络设计的，并且可以在很高的移动率下很好地工作。

# 第二章 代码介绍

## 2.1 符号与缩写

|  |  |
| --- | --- |
| 符号 | 说明 |
| DSR  ACK  OPT  IO  PKT  RERR  RREP  RREQ  SRT  NEIGH | 动态源路由  确认  选项  发送与接收  分组/包  路由错误  路由应答  路由请求  源路由  邻居节点 |

表2.1 DSR协议符号缩写说明

## 2.2 文件介绍

|  |  |
| --- | --- |
| 文件名 | 说明 |
| debug.h  debug.c  ack.h  ack.c  io.h  io.c  opt.h  opt.c  pkt.h  pkt.c  rerr.h  rerr.c  rrep.h  rrep.c  rreq.h  rreq.c  srt.h  srt.c  dsr.h  endian.c  neigh.h  neigh.c  timer.h | 用于检查运行时错误的函数与变量定义  用于检查运行时错误的函数与变量实现  用于路由维护的函数定于  用于路由维护的函数实现  用于发送与接收分组函数的定义  用于发送与接收分组函数的实现  定义了DSR选项与选项头的结构和对DSR选项头与选项的定于与实现  主要定义了对分组的操作函数  主要实现了对分组的操作函数  主要定义了路由错误选项，路由错误的发送与接收函数的定于与实现  主要定义了路由应答选项的结构和对路由应答表操作的函数的定义与实现  主要定义了路由请求选项的结构和对路由请求表操作的函数的定义与实现  主要定义了源路由与源路由选项的结构  主要实现了源路由的处理函数  整个代码中的宏定义  主函数  对邻居表操作函数的定于  对邻居表操作函数的实现  定时器 |

表2.2 DSR 路由协议主要文件

## 2.3 主要结构体定义

|  |  |
| --- | --- |
| 结构体 | 说明 |
| dsr\_ack\_req\_opt  dsr\_ack\_opt  dsr\_pkt  dsr\_rerr\_opt  dsr\_rrep\_opt  dsr\_rreq\_opt  dsr\_srt\_opt  dsr\_srt | 确认请求选项  确认选项  DSR分组/DSR包  路由错误选项  路由应答选项  路由请求选项  源路由选项  源路由 |

表2.3 DSR中的主要结构体定义

# 第三章　DSR工作流程

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。

## 3.1系统流程图

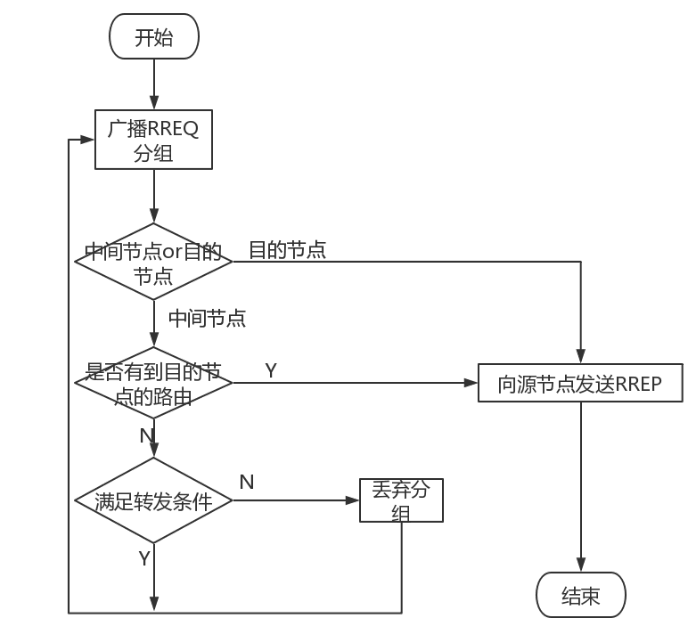


图3.1 DSR系统流程图

## 3.2 路由发现

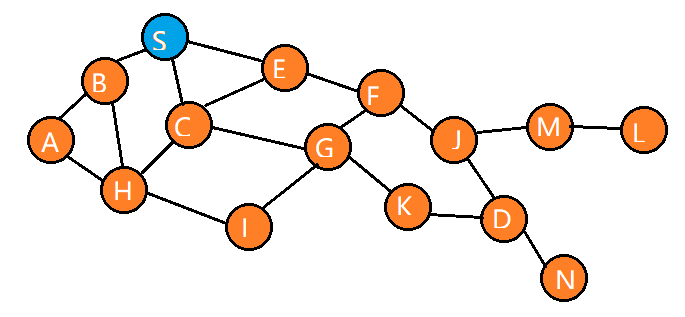


图3.2.1

考虑在图3.2.1的网络拓扑图中，假设节点S要发送数据给节点D，但是检查路由缓存后发现不知道S->D的路由信息，于是S以洪泛的方式向邻居节点广播路由请求RREQ。

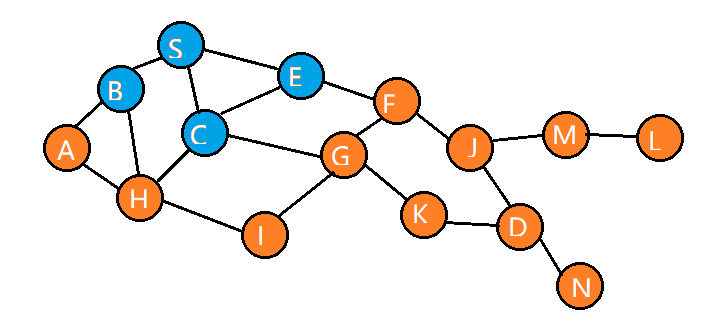


图3.2.2

节点B，C，E收到RREQ后查询自身路由缓存表，未发现相关路由信息，于是将自身地址信息加入数据分组Header部分后继续向邻居节点广播。

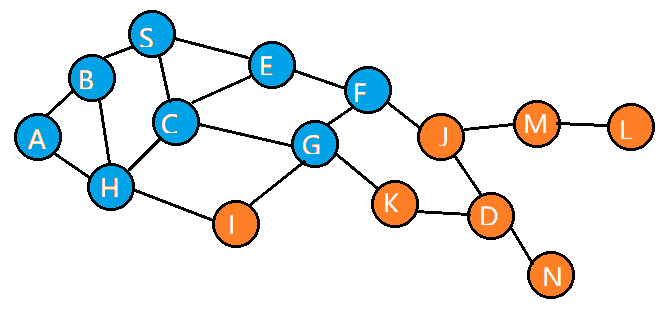
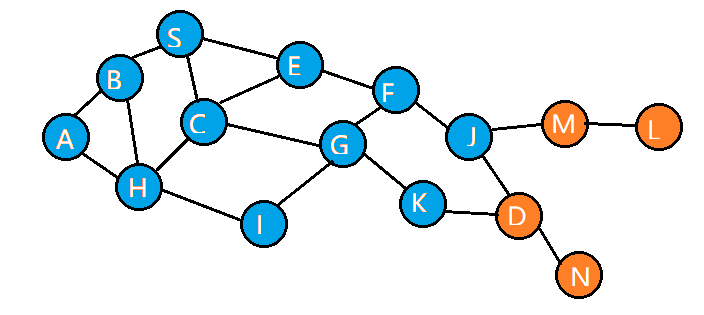


图3.2.3

继续上述过程，其中C节点从G节点收到数据分组后不会继续转发，而是选择无视掉，因为C节点之前已经转发过这个路由请求。



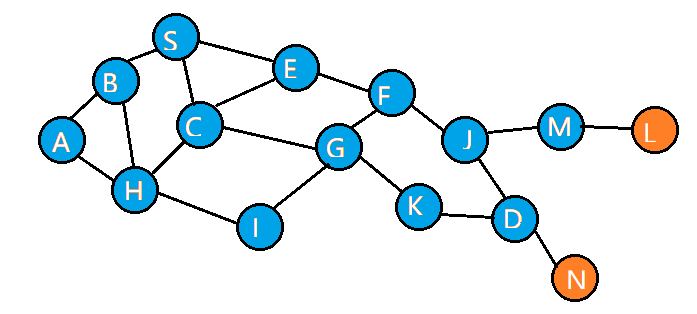


图3.2.4

目的节点D收到路由请求后发现自身是目的节点，不再继续转发，而是返回路由回复route reply（RREP）数据分组。

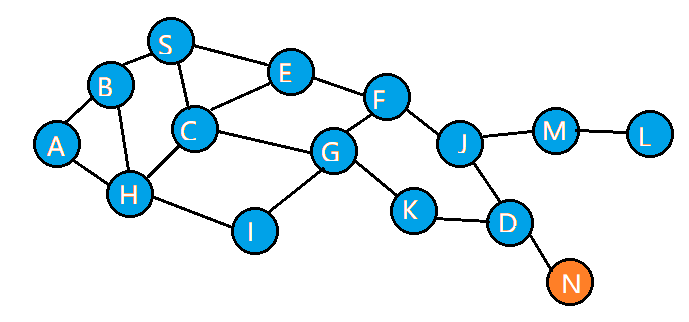


图3.2.5

随后S存储该路由信息，每次发送的数据分组的头部都会包含相应的路由信息，从而可以正常通信。

## 3.3 路由维护

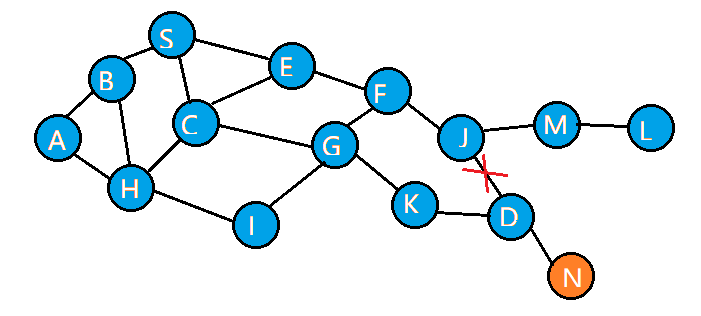
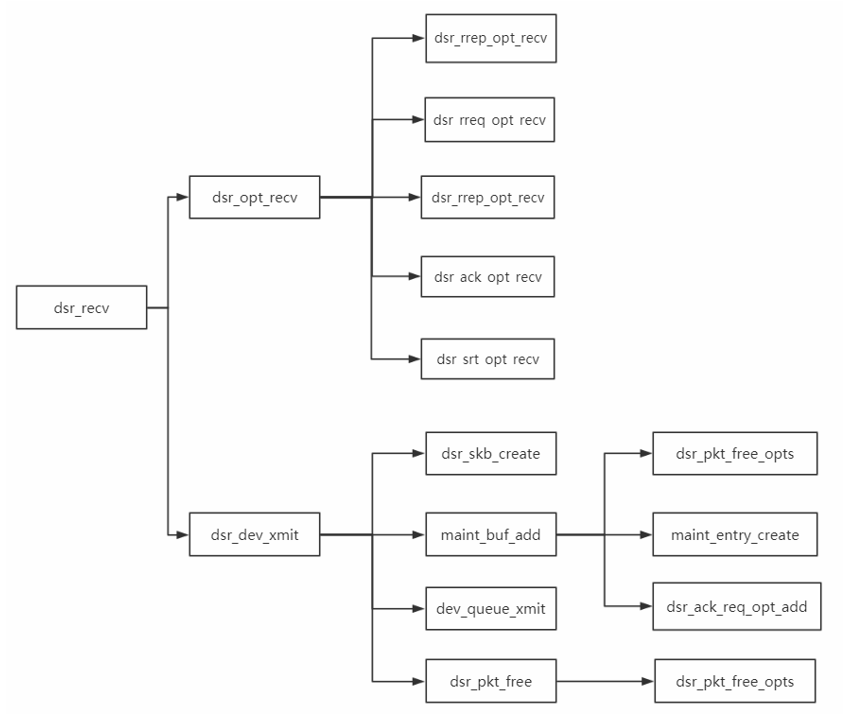


图3.3.1

假设某个时刻，节点J，D之间断开连接，当J节点试图向节点D转发数据分组时，就会向源节点S返回一个路由错误RERR，收到这个错误信息的节点会相应地更新自己的路由缓存信息。

## 3.4 函数调用图



# 第四章 DSR 路由协议的实现

## 常规包处理

### 4.1.1初始化一个包（分组）

首先，在该节点的路由缓存中搜索数据包报头IP目标地址字段中给定地址的路由。

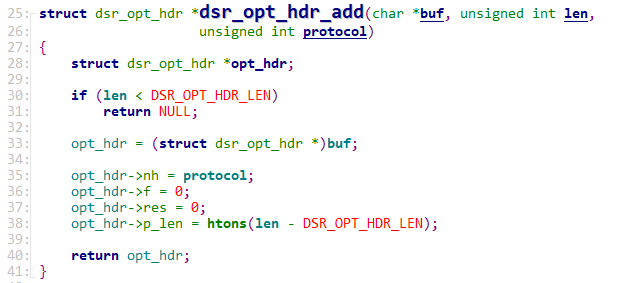
如果在路由缓存中没有找到这样的路由，则对目标地址执行路由发现。为目标结点地址发起一个 路由发现会造成添加一个路由请求选项在这个现有的包,或者保存现有的数据包到发送缓冲区并且通过发送一个单独的包包含这样一个路由请求的选项启动路由发现。如果是前者,它将取代IP目的地址字段为IP的广播地址(255.255.255.255),将原始IP目的地址复制到新的路由请求的目标地址字段选项添加到包。

如果包现在不包含路由请求选项，则该节点必须具有 到包的 目的地地址的路由;如果节点有不止一条到该目的地地址的路由，则节点将为该包选择一条路由。如果这条路线的长度大于1跳,或者节点决定在这条路由的第一跳请求一个DSR网络层的ACK,然后插入一个DSR选项头 包,并且插入一个DSR源路由选项。

将数据包发送到选定源路由中给定的第一跳节点地址，利用路由维护确定下一跳的可达性。

### 4.1.2将DSR选项头添加进包中

将DSR选项头添加到数据分组代码实现如下：



28-40：首先定义每个选项头的固定部分，之后判断如果长度足够，则开始装载相关设置，将参数中缓冲区指针强制类型转换为一个指向DSR选项头的指针，Next Header字段设置为数据分组的IP头的协议号，最后返回该指针。

具体分析：

发起包的节点在必要时向包添加DSR选项头，以携带路由协议所需的信息。一个包不能包含一个以上的DSR选项头。DSR选项标头通过执行以下步骤添加到包中:

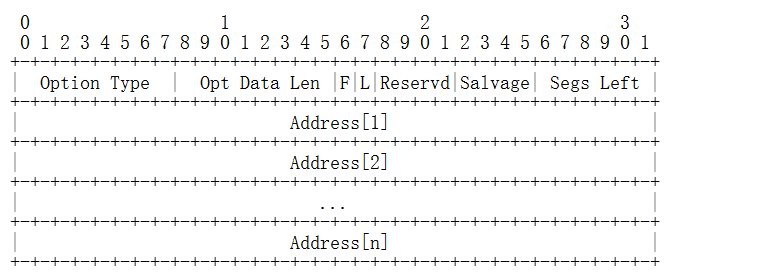
1.在IP报头后面插入DSR选项报头，但要在可能出现的任何其他报头之前插入。

2.将DSR选项报头的下一个报头字段设置为包的IP报头的协议号字段。

3.将数据包IP报头的协议字段设置为分配给DSR的协议号。

### 4.1.3将DSR源路由选项添加进包中

结构图：

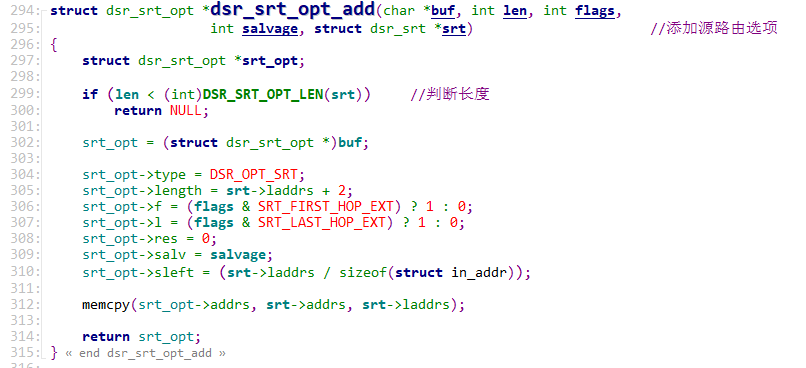


源码如下：



23-42：sleft记录剩余路段的数量，即在到达最终目的地之前仍然需要访问的中间节点数量。变长数组addrs记录源路由的路径信息。其他字段同上。

发起分组的节点在需要时向分组添加DSR源路由选项，以便将来自该始发节点的源路由携带到分组的最终目的地地址，代码实现如下：



297-312：选项头固定部分定义，缓冲区长度判断以及指针转换同上，接下来开始装载选项字段，将类型设置为源路由选项，初始化长度，标志位，抢救次数，segmens left，随后数据分组中的地址被复制到DSR源路由选项中的addrs数组字段。

具体分析：

如果需要，发起信息包的节点向信息包添加DSR源路由选项，以便将源路由从该发起节点传送到信息包的最终目的地地址。具体来说，添加DSR源路由选项的节点构造DSR源路由选项，并按照以下步骤顺序修改IP包:

1. 节点创建DSR源路由选项，并将其追加到包中的DSR选项头。

2. DSR源路由选项(n)中包含的Address[i]字段的数量是包源路由中的中间节点的数量。DSR源路由选项中的段左字段初始化为n。

3. 数据包源路由中的地址被复制到DSR源路由选项中的顺序地址[i]字段中，其中 i = 1,2，…,n。

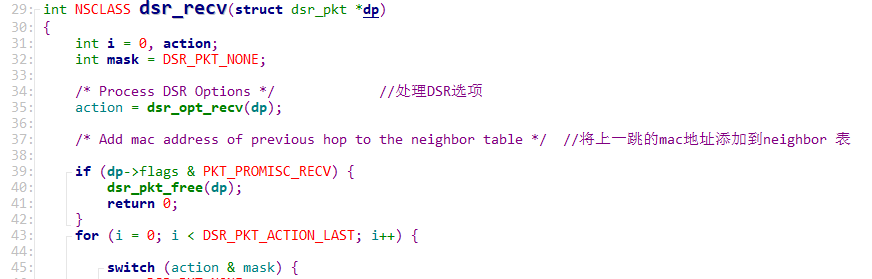
4. DSR源路由选项中的(F)位（First Hop External）是从标记包源路由中第一跳的外部位复制的，如路由缓存中所示。

5. DSR源路由选项中的(L)位（Last Hop External）是从标记包源路由中最后一跳的外部位复制的，如路由缓存中所示。

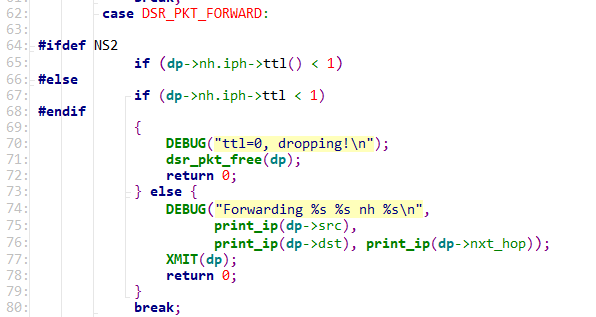
6. DSR源路由选项中的打捞字段（补救字段）初始化为0。

### 4.1.4处理一个接收到的包

当一个节点收到任何数据分组时，如果该数据分组包含一个DSR选项头，那么该节点必须处理该DSR选项头中包含的所有选项。处理路由请求的代码实现如下：

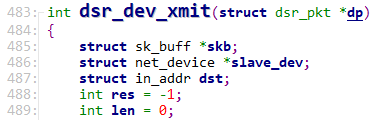


31-43：节点会根据数据分组携带的选项来进行不同的处理操作，调用的dsr\_opt\_recv函数通过对数据分组中选项的类型进行识别然后返回对应的操作类型，随后使用switch语句根据返回的操作类型进行不同的操作，在这些分支中这里着重分析较为复杂的转发请求，代码如下：

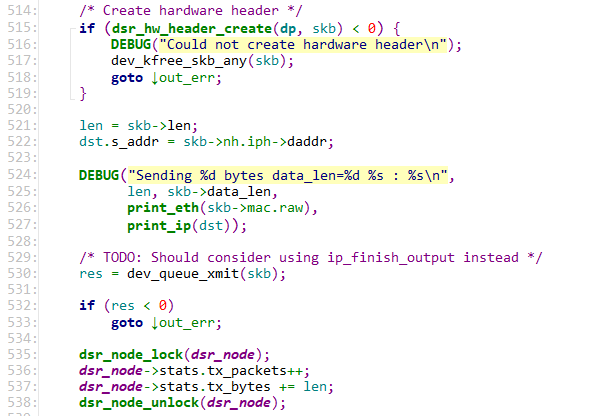


首先进行生存期ttl的判断，如果 ttl为零则丢弃数据分组，不为零则调用XMIT函数进行发送数据，XMIT是如下函数dsr\_dev\_xmit的宏定义，代码如下：

（该函数实现了dsr数据包到IP数据包的向下封装）



其中，结构体sk\_buff与m\_buf类似，是linux系统中各层协议之间传递数据的重要载体，net\_device结构体用来描述一个网络设备，dst字段标记目标地址：



497-507：通过锁来实现dsr\_node（描述当前节点信息）的并发安全性，将抢救网卡设备指定为当前节点的相应设备字段，随后调用dsr\_skb\_create函数（装填相关字段信息并返回一个sk\_buff的指针）创建数据分组。

515-538：随后开始向创建后的数据分组中装填mac地址相关字段，并调用dev\_queue\_xmit函数进行发送。

具体分析：

当某个结点接收到一个包时，如果这个结点包含DSR选项头，该结点必须处理所有包含在该DSR选项头中的选项，步骤如下：

如果选项头中包含一个路由请求选项（路由发现），节点应该从路由请求中提取源路由，并将该路由信息添加到自己的路由缓存中。路由请求中的路由信息序列是地址发起者、地址[1]、地址[2]、…,地址[n] 。每个地址是一个节点通过这条路线请求已经通过了结点。这里的值n是路由请求选项中记录的地址数。

在可能更新节点的路由缓存以响应路由请求选项中的路由信息之后，节点必须按照2.2节所述处理路由请求选项。

如果DSR选项头包含路由应答选项，节点应该从路由应答中提取源路由，并将该路由信息添加到其路由缓存中。

在可能更新节点的路由缓存以响应路由应答选项中的路由信息之后，如果包的IP目的地地址与该节点的某个IP地址匹配，则节点必须按照2.6节所述处理路由应答选项。

如果DSR Options报头包含Acknowledgement选项，节点应将从ACK源地址字段标识的节点到ACK目的地地址字段标识的节点的单个链路添加到路由缓存。

在可能响应于确认选项中的路由信息更新节点的路由缓存之后，节点必须接着处理3.3节中描述的确认选项。

如果DSR选项头包含DSR源路由选项，节点应该从DSR源路由中提取源路由，并将该路由信息添加到其路由缓存中。如果打捞字段的值在DSR源路由选项是零,然后从DSR路由信息的源路由的顺序跳地址,如果打捞非零,从DSR路由信息来源途径是跳地址的顺序的值是在源中的源地址字段IP报头的包携带DSR源路由选项(原始数据包的发送者),每个地址[i]是DSR源路由中的地址[i]字段的值，destination是包的IP报头中的目标地址字段的值(源路由的最后一跳地址)。这里的值n是DSR源路由选项中源路由中的地址数。

最后，如果数据包IP报头中的目标地址与该接收节点自己的IP地址(es)匹配，则删除DSR选项报头和报头中包含的所有DSR选项，并将剩余的数据包传递到网络层。

### 4.1.5处理收到的DSR源路由选项

当一个节点收到一个包含DSR源路由选项的数据包时，该节点应该检查数据包以确定是否可以进行自动缩短路由，具体来说，如果一个节点不是该数据分组的下一跳地址，但是在路由信息的后续地址中有该节点的地址，则应执行自动路径缩短，代码实现如下：



432-439：判断是否可以进行路由缩短：如果节点自身地址并非下一跳地址，并且在路由后续地址中查找到了自己的地址的同时自身应答列表中尚没有该路由回复，则调用dsr\_srt\_shortcut函数进行路由缩短，返回一个缩短后的源路由选项结构src\_cut。

447-461：如果缩短后剩余节点个数为0（即自身为目的节点），则直接调用dsr\_srt\_new\_rev函数将src\_cut中的路径进行反转（因为要向源节点发送应答，需原路返回），如果自身不是目的节点，则调用dsr\_srt\_new\_split\_rev函数，该函数会先调用dsr\_srt\_new\_split函数进行分割（切割掉多余的路径），将路径信息切割至自身节点后再调用dsr\_src\_new\_rev函数进行反转。随后该节点会使用反转后的路由信息向源节点发送应答分组，该分组携带有源节点到该节点的缩短后的路由信息。

如果不能缩短则直接转发。

具体分析：

当节点接收到包含DSR源路由选项的包时(无论是转发、窃听还是作为包的最终目的地)，该节点应检查包，以确定收到该包是否意味着自动缩短路由的机会。具体来说,如果这个节点不是这个包的下一跳但后来被命名为不可避免的部分源路由数据包的DSR源路由选项,那么说明这个包有自动路线缩短的机会。在这种情况下，该节点应该执行以下步骤序列，作为自动缩短路由的一部分:

1. 节点搜索其免费路由应答表（Gratuitous Route Reply），以查找描述该节点早先发送的无偿路由应答的条目，触发无偿路由应答的分组的原始发送方和该节点为了触发无偿路由应答而从其窃听该分组的发送节点，两者都匹配这个新接收的数据包的节点地址。如果在节点的免费路由应答表中发现这样的条目，则节点不应该执行自动路由缩短。

2. 否则，节点将在其免费路由应答表中为该窃听包创建一个条目。这个新条目的超时值应该初始化为GratReplyHoldoff。此超时过期后，节点应从其免费路由应答表中删除此条目。

3. 在创建上述新的免费路由应答表条目之后，节点将对该窃听包的IP源地址发起免费路由应答。

4. 如果在网络中使用的MAC协议不能够通过单向链路传输单播分组，那么在发起此路由应答时，节点必须使用源路由来路由应答分组，该路由应答分组是通过反向触发无偿路由应答的分组被路由到的跳序列而获得的。

5. 丢弃偷听到的包，因为包是在正常遍历包的源路由之前收到的，这将导致它到达这个接收节点。该包的另一个副本通常会按照该包的源路由指示到达该节点;丢弃包的这个初始副本将防止这个包的复制。

如果上述自动路由缩短过程中没有丢弃数据包，则节点必须按照以下步骤顺序处理源路由选项:

1.如果DSR源路由选项中左字段的段的值等于0，那么从DSR选项头中删除DSR源路由选项。

2.否则，让n等于(Option Data Len - 2) / 4。这是DSR源路由选项中的地址数。

3.如果段左字段值大于n，则发送一个ICMP参数问题（ICMP Parameter Problem），代码0，消息[31]到IP源地址，指向段左字段，丢弃数据包。不要进一步处理DSR源路由选项。

4.否则，将Segments Left字段的值减1。让i等于n减去左字段（Segments Left）。这是地址向量中要访问的下一个地址的索引。

5.如果地址[i]或IP目标地址是组播地址，则丢弃数据包。不要进一步处理DSR源路由选项。

6.如果这个节点有多个网络接口，并且Address[i]是这个节点的一个网络接口的地址，那么这表示在转发数据包时使用的网络接口发生了变化，如第8.4节所述。在本例中，将段左字段的值减1，以跳过这个地址(表示网络接口更改)，并转到上面的第一步(检查段左字段的值)，继续处理这个源路由选项;在进一步处理这个源路由选项时，必须使用指定的新网络接口转发包。

7.如果该链路上传输给地址[i]是最大传输单元（MTU）小于包的大小，这个节点要么选择丢弃包并且发送一个信息为Packet Too Gig的ICMP包给这个包的原地址[31]，要么破译它。

8.将数据包转发到IP报头地址[i]字段中指定的IP地址，遵循正常的IP转发过程，包括检查和减少数据包IP报头的Time-to-Live (TTL)字段。在包的这种转发中，下一跳节点必须被视为一个直接邻居节点:到下一个节点的传输必须在一个IP转发跳中完成，不需要路由发现，也不需要搜索路由缓存。

9.在转发包时，通过验证下一跳节点是可到达的，对包的下一跳执行路由维护。

多播地址不能出现在DSR源路由选项中，也不能出现在在DSR选项头中带有DSR源路由选项的包的IP目标地址字段中。

## 4.2 路由发现

路由发现发生在当网络中的某个源节点S要向目的节点D发送数据并且S不知道通向D 的路由时，目的是帮助源节点获得到达目的节点的路由。

当源节点S需要向目的节点Ｄ发送数据时，它首先查询路由缓冲区是否有通向Ｄ的路由，如果有则将该路由信息附加到数据分组头部中进行发送，如果没有，则发起路由发现过程。源节点向邻居节点以洪泛方式发送路由请求，以此启动路由发现，随后每个中间节点接收后判断是否存储有到目的节点的路由，若不含有，则将自己的地址加入请求数据中的路由记录并继续转发给邻居节点（如果是目的节点D收到该请求，则会返回路由回复，路由回复中包含所请求的路径信息）。当源节点收到路由回复后，将收到的路径信息存储在自己的路由缓存中，路由发现过程结束。

在扩展的路由发现过程里，每个节点还可以从转发的数据分组头部中学习到路径信息并保存在自己的路由缓存中。同时如果中间节点在自己的路由缓存中找到了源节点请求的路由信息，也会返回一个路由回复而不是继续转发该路由请求。

### 4.2.1一个例子

结点A，节点B。每个路由请求标识路由发现的发起者和目标，并且还包含由请求发起者确定的惟一请求标识（设为2）。每个路由请求还包含一个记录，其中列出转发路由请求的这个特定副本的每个中间节点的地址。此路由记录由路由发现的发起者初始化为一个空列表。在本例中，路由记录最初只列出节点A。

当另一个节点接收到此路由请求(如本例中的节点B)时，如果它是路由发现的目标，它将向路由发现的发起者返回一个“路由应答”，并从该路由请求中获得累计路由记录的副本;当发起者收到此路由应答时，它将此路由缓存在其路由缓存中，以便在将后续数据包发送到此目的地时使用。

否则，如果接收路由请求的此节点最近看到来自此发起者的另一条路由请求消息，该消息带有相同的请求标识和目标地址，或者如果该节点自己的地址已经在路由请求中的路由记录中列出，则该节点将丢弃该请求。(节点会考虑最近看到的请求，如果它在路由请求表中仍然有关于该请求的信息。否则，该节点将自己的地址附加到路由请求中的路由记录，并通过将其作为本地广播包(具有相同的请求标识)进行传播。本例中，节点B广播节点C接收到的路由请求;节点C和D也依次广播请求，导致节点E接收到请求的副本

返回路线回复发起者的路由发现,比如在这个例子中,节点E回复回节点,节点E通常会检查自己的路由缓存路线回,如果发现,将使用它的源路由数据包包含路由应答的交付。否则,E应该履行自己的目标节点的路由发现,但为了避免可能的无限递归的路由发现,它必须通过包包含这条路线回复自己的路线要求A .也可以捎带其他小数据包,比如TCP SYN包,在路由请求使用相同的机制。

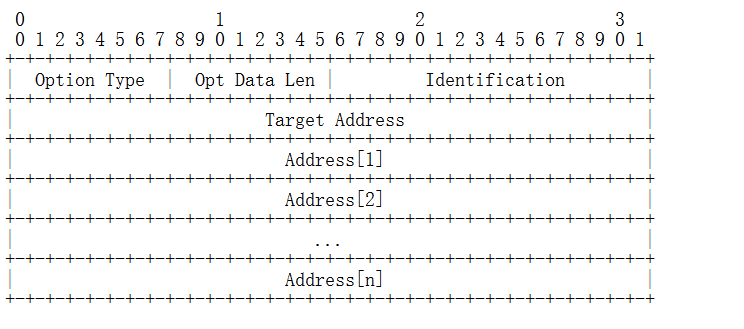
相反，节点E可以简单地反转它试图在路由应答中发送的路由记录中的跃点序列，并将其用作承载路由应答本身的包上的源路由。

在启动路由发现时，发送节点将原始数据包(触发发现的数据包)的副本保存在称为“发送缓冲区”的本地缓冲区中。发送缓冲区包含这个节点不能传输的每个包的副本，因为它还没有到包的目的地的源路由。发送缓冲区中的每个包在逻辑上与它被放入发送缓冲区的时间相关联，并在驻留在发送缓冲区一段超时时间后被丢弃;如果需要防止发送缓冲区溢出，也可以使用FIFO或其他替换策略在数据包过期之前将其逐出。

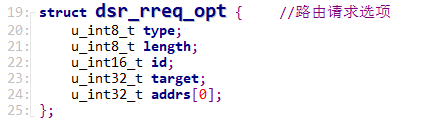
### 4.2.2 发起路由请求

为某个目标发起路由发现的节点在某个IP包的DSR选项头中创建并初始化路由请求选项。

路由请求选项结构图：



源码如下：



20-21：选项通用部分，选项类型和长度，如上所述。不理解此类型的节点会忽略此选项。

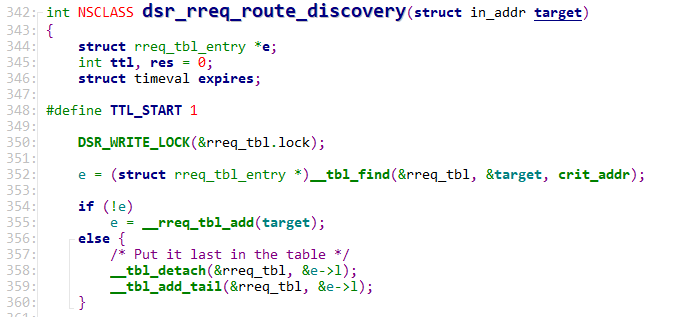
22：id由路由请求发起者设置，以此作为每个请求的唯一标识，同时id值也使得每个接收节点可以识别它是否已经接收过同一请求，如果接收节点在自己的路由请求表中发现了该id值（对于特定的源IP地址和目的IP地址该id值是唯一的），那么这个节点就会忽略本次路由请求。

23：target记录路由请求的目的节点地址

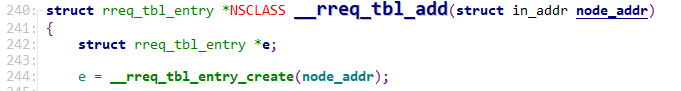
24：addrs变长数组用以记录路由请求传播路径上的每个节点的地址信息。地址数量由该选项中的length指定，id和target共占6个字节，每个ip地址4个字节，故n=（length-6）/4

路由请求选项不得在一个DSR选项头中出现超过一次。

启动路由发现的节点会在某个IP数据包的DSR选项头中创建并初始化一个路由请求选项。这可以是一个单独的IP数据包，仅用于执行此路由请求选项，也可以在某些需要发送到目标节点的现有数据包中包含路由请求选项。需要完成的步骤有1添加到本地路由请求表2发送路由请求。代码实现如下：



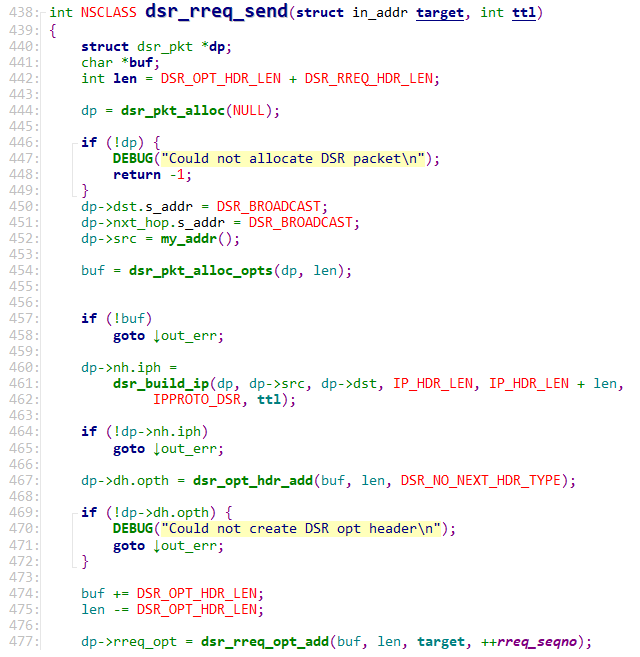
352-360：首先通过\_tbl\_find函数在路由请求表中查找是否有相同目的地的路由请求，如果没有则调用\_\_rreq\_tbl\_add函数直接添加进路由请求表，否则先删除原有记录，再调用\_tbl\_add\_tail添加进路由请求表中的最后（使用先进先出的管理策略，如果请求表满会先删除第一条记录）。其中\_\_rreq\_tbl\_add函数主要调用\_rreq\_tbl\_entry\_create函数，代码如下：





206-235：首先为该条请求记录申请内存空间，然后开始装填状态，目标地址，生存期ttl等字段。最后需要为每条请求初始化一个唯一标识id，避免被无限转发。

发送请求部分代码如下：



C:\Users\lenovo\AppData\Roaming\Tencent\Users\531973083\QQ\WinTemp\RichOle\)`KJM[V6RRTJ%J_3F]C{Z2N.png

440-491：首先申请构造一个DSR数据包，将源地址设置为自身地址，目的地址和下一跳地址都设置为广播地址，然后调用dsr\_build\_ip函数构造IP报头，调用dsr\_opt\_hdr\_add添加DSR选型头，再调用dsr\_rreq\_opt\_add函数向DSR选项头中添加路由请求选项，最后调用XMIT函数发送数据分组。

具体分析：

这可能是一个分离出的IP包,只用来承载这条路线请求选项。或可能选择在一些现有的包 包括该路由请求,这些包需要发送到目标节点。路由请求选项必须包含在包中的DSR选项头中。为了初始化路由请求选项，节点执行以下步骤序列:

1.选项中的选项类型必须设置为值2。

2.选项中的Opt Data Len字段必须设置为值6。路由请求选项初始化时的总大小为8个字节。

3.选项中的标识字段必须设置为一个新值，不同于此节点最近为相同目标地址发起的其他路由请求。例如，每个节点可以维护一个计数器值，以便为它发起的每个路由请求生成一个新的标识值。

4.选项中的目标地址字段必须设置为此路由发现的目标IP地址。

5.这个包的IP报头中的源地址必须是节点自己的IP地址。这个包的IP报头中的目的地址必须是IP广播地址(255.255.255.255)。

节点必须在其路由请求表中维护其发起的路由请求的信息。当发起一个新的路由请求时，节点必须使用路由请求表条目中记录的该路由请求目标的信息，并且必须更新表条目中的该信息，以便在为该目标发起的下一个路由请求中使用。特别的：

目标节点的路由请求表条目记录了该节点为该目标节点发起的最后一次路由发现的路由请求的IP头中使用的Time-to-Live (TTL)字段。该值允许节点实现各种算法，以控制其路由请求在为目标发起的每个路由发现上的传播。

节点必须使用退避算法，以限制该节点对相同目标地址发起新路由发现的速率。特别地，直到接收到针对该目标节点地址的有效路由应答，具有相同跳数限制的目标节点的连续路由发现发起之间的超时应该通过在每个新发起上加倍超时值来增加。

节点使用DSR源路由选项和路由请求选项处理包含DSR选项头的包的行为是未指定的。包不应该同时包含DSR源路由选项和路由请求选项。包含路由请求选项的包不应包含确认请求选项，不应期望链路层确认或被动确认，也不应重新传输。

### 4.2.3 处理接收到的路由请求选项

当一个节点收到一个包含路由请求选项的数据包时，该节点的操作如下：

首先判断是否满足丢弃条件或返回条件

1如果路由请求中的目标地址字段与这个节点自己的IP地址相匹配，那么节点应该返回一个路由回复给这个路由请求的发起者

2如果不匹配，节点必须检查路由请求选项（IP源地址字段和地址字段序列）中记录的路由，以确定此节点自己的IP地址是否已经出现在该地址列表中。如果查找到（即自己已经处理过），则该节点会丢弃整个分组。如果未查找到，则该节点会在其路由请求表中搜索该路由请求的发起者（IP源地址字段）的记录。

3如果在表中找到这样一个记录，那么节点必须搜索该表项中最近接收到的路由请求的标识值缓存表，以确定在该路由中是否存在与该标识值和目标节点地址匹配的高速缓存中的条目请求。如果在这个缓存中发现这样一个（标识，目标地址）条目在路由请求表中（即自己已经接收过），那么节点必须丢弃整个携带路由请求选项的分组。

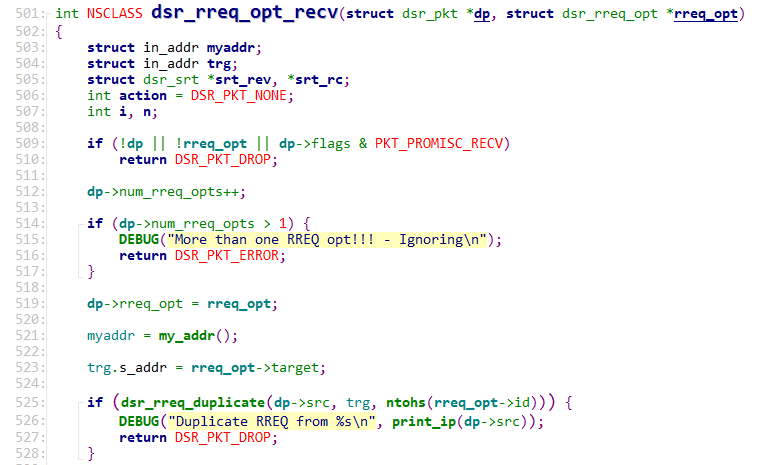
判断完毕后如果未丢弃或是返回，则这个节点应该按照以下步骤进一步处理路由请求：

1在最近收到的路由请求的缓存中为此路由请求添加一个条目。

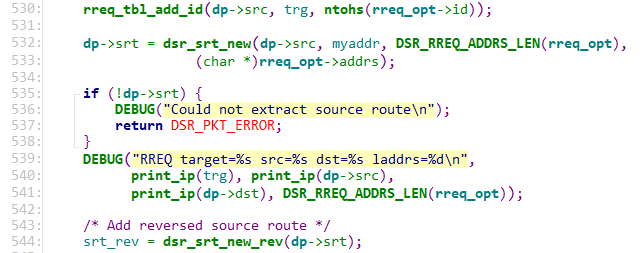
2将此节点自己的IP地址追加到路由请求中的地址值列表中（即addr数组），并将路由请求中的Opt Data Len字段的值增加4（IP地址的大小）。

3同时这个节点会搜索自己的路由缓存，查找是否有到这个目标地址的路由。如果找到，则向这个路由请求的发起者返回路由答复。如果节点未找到相关路由，那么这个节点会把这个数据包的副本作为链路层广播发送，

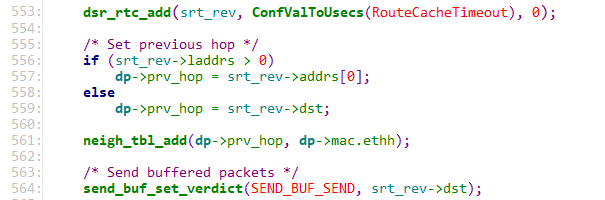
代码如下：



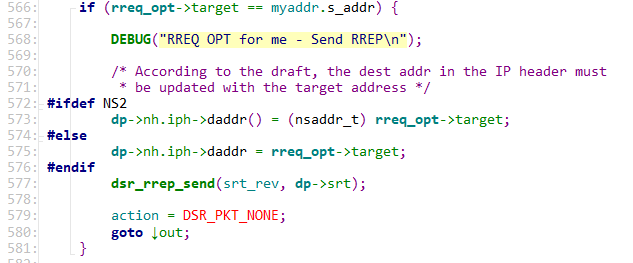
503-527：这里调用dsr\_rreq\_duplicate函数判断是否收到的路由请求是否已经在自己的路由请求表中，如果已经包含则丢弃该请求包。



532-544：先将该请求添加进路由请求表中并关联该请求的标识id，再新建一个源路由选项，随后调用dsr\_srt\_new\_rev函数将所收到的数据包中的路径信息反转（参考3.4.1），因为之后可能需要返回路由回复信息。



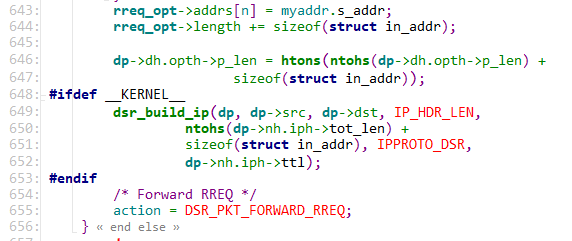
553-564：先调用dsr\_rtc\_add函数将该路由信息添加进自身的路由缓存中（节点可以从路由请求听到路由信息），之后如果剩余地址数大于0则将自身节点地址最后将该数据分组复制到发送缓冲区中。



566-580：如果请求的目标地址是自身地址，则将IP Header字段设置为自身地址，随后该节点调用dsr\_rrep\_send函数返回路由回复。



595-625：接着在自身的路由缓存表中查找该目的地址的路由，如果找到，就将请求中的源路由部分和查找到的路由部分拼接起来，在检查确认拼接后的路由没有重复节点后，调用dsr\_rrep\_send函数使用之前反转后的路径发送该路由回复信息（携带拼接后的路径）。



643-655：如果拼接后的路由信息包含重复节点，则不得向源节点发送路由回复，而是选择将自身节点地址添加到addrs数组中，构造IP数据报头，之后转发该数据分组。

### 4.2.4 使用路由缓存生成路由应答（“缓存路由应答”）



详细分析：

如果节点在其路由缓存中有一条从自身到该目标的路由，则节点可以处理接收到的路由请求，以避免将路由请求进一步传播到该请求的目标。节点从自己的缓存路由生成到路由请求目标的路由应答称为“缓存路由应答”，该机制通过减少路由请求的泛滥，可以大大降低网络上路由发现的总开销。接收到的路由请求的一般处理在2.2节中描述。

在处理接收到的路由请求时，为了可能返回缓存的路由响应，节点必须在其路由缓存中具有从自身到该路由请求目标的路由。但是，在为该路由请求生成缓存路由应答之前，节点必须验证在该路由请求中累积的路由中没有列出重复地址，以及来自该节点的路由缓存的路由。如果这些地址之间存在重复，则节点不能发送缓存的路由应答。节点应该继续处理2.2节中描述的路由请求。

如果路由请求和来自路由缓存的路由满足上述限制，则节点应构造并返回缓存的路由应答如下:

回复的源路由是hop地址的序列,其中发起者是这个路由请求的初始化器的地址,每个地址[i]是路由请求的地址,c路由（c-route）是从节点的路由缓存获得的从源路由到这个目标节点的hop地址的顺序。在将此缓存的路由附加到源路由时,必须排除该节点本身的地址,因为它已经被列出为地址[n]。

使用2.4中定义的过程,向路由请求的发起者发送路由应答。路由请求的发起者在数据包的IP头的源地址字段中表示。

然而,在发送缓存路由应答之前,节点可能会延迟回复,以帮助防止可能的路由回复“风暴”。

如果节点返回一个缓存的路由应答,如上所述,则节点不能进一步传播该路由请求。也就是说，如果包中没有其他DSR选项,在DSR选项头之后没有有效负载,那么节点应该简单地丢弃数据包。否则，如果包包含其他DSR选项,或者在DSR选项头之后包含任何有效负载,节点应该将包沿着缓存的路径转发到路由请求的目标结点。具体步骤如下：

1.将目标地址从DSR选项头的路由请求选项复制到包的IP头中的目标地址字段。

2.从数据包中的DSR选项头删除路由请求选项,并在包的DSR选项头中添加DSR源路由选项。（将路由请求变为源路由）

3.在DSR源路由选项中,设置地址[i]字段去表示在该节点的路由缓存中发现的源路由到路由发现的原始目标。具体来说,该节点将源路由的hop地址复制到DSR源路由选项的顺序地址[i]字段中,为i = 1,2,…，地址[1]这里是这个节点本身的地址(从这个节点找到的源路由的第一个地址是路由发现的原始目标)。这里的值n是这个源路由中hop地址的数量,不包括包的目的地(它已经在包的IP头中的目标地址字段中表示)。

4.初始化DSR源路由选项中段左字段为n,如上面所定义。

5.DSR源路由选项中的(F)位（First Hop External）是从标记包源路由中第一跳的外部位复制的，如路由缓存中所示。

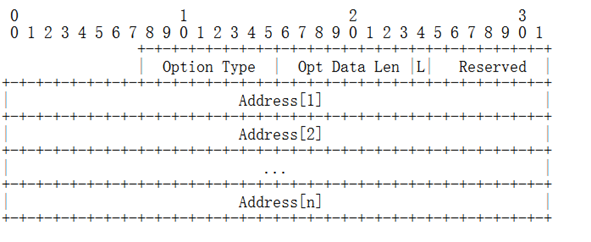
6.DSR源路由选项中的(L)位（Last Hop External）是从标记包源路由中最后一跳的外部位复制的，如路由缓存中所示。

7.DSR源路由选项中的打捞字段必须初始化为某个非零值;使用的非零值应该是MAX\_SALVAGE\_COUNT。通过将该字段初始化为非零值，转发或无意中听到此包的节点将不会认为在DSR源路由选项中的该包的IP源地址和地址[1]地址之间存在链接。通过选择MAX\_SALVAGE\_COUNT作为节点初始化此字段的非零值，节点将不会尝试回收此包。

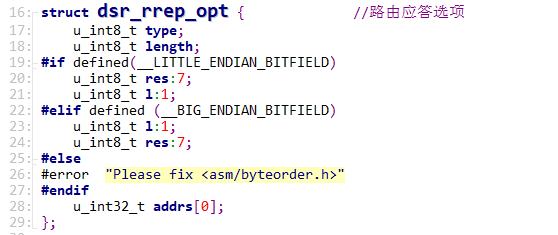
8.使用第1.5节中描述的转发过程,将数据包传输到包中的新源路由上的下一个跳跃节点。

### 4.2.5发起路由应答

路由应答结构图：



源码如下：

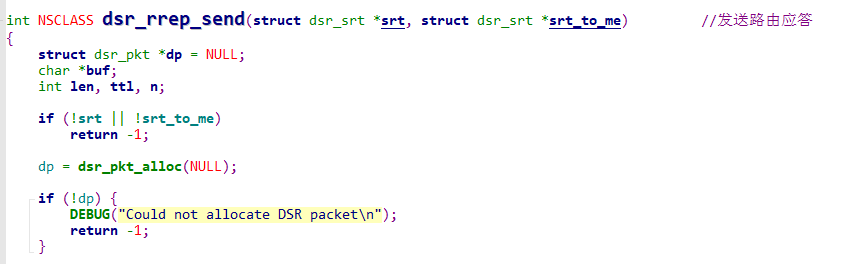


17-28：根据系统大小端的区别，定义不同的位域作为标志位。Res表示保留位，置0。L标识网络内外部，置1时，表明路由回复中的最后一跳到达了DSR网络的外部。变长数组addrs记录返回的路由信息。同样，其地址数量由length字段指定。

一个路由回复选项可能会在一个DSR选项头中出现不止一次。

具体分析：

一个结点发起一个路由应答是为了回复一个接受并处理过的路由请求。路由应答在路由应答选项中返回。路由应答选项可以返回一个单独的IP包给该路由请求的发起者,只用作此路由应答选项,或者在发送到源结点的其他IP包中包含它。



路由应答选项必须包含在返回到发起者的包中的DSR选项头中。为了初始化路由应答选项,节点执行以下步骤:

1.选项中的选项类型必须设置为值3。

2.选项中的Opt Data Len字段必须设置为值(n \* 4)+ 3,其中n是返回的源路由中地址的数字(不包括路由发现发起者结点的地址)。

3.在选项中,Last Hop External (L)位必须被初始化为0。

选项中的保留字段必须初始化为0。

必须将路由请求标识符初始化到路由请求的标识符字段,请求响应。

4.源路由中的hop地址序列被复制到选项的地址[i]字段中。地址[1]必须设置为路由发现的发起者结点后路由的第一跳地址,地址[n]必须设置为源路由的最后地址(目标节点的地址)。

5.在传递路由应答选项的包的IP头中,目标地址字段必须设置为路由发现的发起者的地址

在创建和初始化路由应答选项和包含它的IP包之后,发送路由应答。在发送此节点的路由响应(但不是从节点转发路由应答)时,这个节点应该延迟响应,由一个小抖动周期随机选择在0到BroadcastJitter。

当在网络中使用的MAC协议返回任何路由应答时,在网络中使用的MAC协议不能够在单向链接上传输单播包,因此,必须通过在路由请求包中转换hops序列来获得路由应答包的路由,从而在路由请求包中转换hops序列(然后在路由回复中返回)。对返回路由应答的限制使路由应答能够测试这一双方向性的hops序列,防止路由应答的路由回复,除非每一个hops都返回了路由应答(因此,在回复中返回的源路由中的每一个hops)都是双向的。

如果向路由请求的发起者发送路由应答需要执行路由发现,那么路由应答选项必须在包含路由请求的包上。这个piggyback阻止了一个循环,在其中,新路由请求的目标(它本身就是原始路由请求的发起者)必须做另一个路由请求,以返回路由应答。

如果将路由请求发送到路由请求的发起者不需要执行路由发现,那么节点应该发送一个unicast路由响应,以响应它所接收到的每个路由请求,以表明它是目标节点。

### 4.2.6防止路由应答风暴

每个节点延时D发送RREP。

其中 D=H \*（h-1+r）

其中H是每条链路的传播延时h是自己返回的路径长度，即到目的节点的跳数r是0或者1

D与节点到目的节点的跳数成正比，使得到目的节点有最短路径的RREP最先发送

节点将接口设置成混杂模式(promiscuous)，监听是否存在有比自己更短的到目的节点的路径，如果有，则不发送本节点的RREP。

在延迟周期内,这个节点将接收所有的数据包,从该路由发现的发起者中寻找数据数据包,以实现该发现的目标。如果此节点在延迟期间接收到的数据包使用长度小于或等于h的源路径,此节点可以推断路由发现的发起者已经收到了路由回复,给出了同样好的或更好的路由。在这种情况下,这个节点应该取消它的延迟计时器,不应该发送路由应答。

### 4.2.7 处理接收到的路由应答选项

如果接收到的包 包含路由应答，节点应该检查每个数据包的发送缓冲区来确定一个路由数据包的IP目的地址是否存在于节点的路由缓存。如果是，那么应该使用该路由发送数据包，并从发送缓冲区中删除。此过程通用于处理接收到的路由应答选项所需的所有处理。

当使用需要双向链路进行单播传输的MAC协议时，可以通过路由请求的传播发现单向链路。当路由应答通过反向路径发送时，转发节点可能会发现下一跳不可到达。在这种情况下，它必须将下一跳地址添加到黑名单。

## 4.3 路由维护

当网络的拓扑结构发生变化后，如果S通向D的路由记录不再有效，此时DSR协议就会启动路由维护，尝试使用其他路由或者通过路由发现寻找一条新的路由。

在数据进行发送的过程中，如果路由维护发现网路拓扑结构发生改变使得某条使用中的路由无效，则会发送路由错误给源节点，源节点会在收到该报文后从路由缓存中删除所有的相关错误路由信息，并重新发起路由发现。

在扩展的路由维护过程中，如果一个要转发数据分组的中间节点通过路由维护检测到该数据分组的路径下一跳已经中断时，该节点会检查其路由缓存，如果发现有另一条路由可以到达该数据分组的目的地，节点就会抢救数据分组而不是直接丢弃。同时数据分组维护一个计数值来标记被抢救次数，以避免被无限抢救。当某个源节点收到它发起的数据分组的路由错误后，它会在下一个路由请求上附加上这个错误信息传播给邻居结点。并且，如果路由中的一个或多个中间节点不再需要，则可以自动缩短使用中的源路由，即自动路线缩短。

### 4.3.1 使用链路层ACK

如果使用中的MAC协议提供了成功交付数据包的反馈(例如由IEEE 802.11[13]定义的链路层确认框架提供的反馈)，那么就没有必要使用DSR确认请求和确认选项。如果链路层反馈可用，它将代替该路由维护中其他任何确认机制，节点不应将被动确认或网络层ACK用于路由维护。

在使用链路层确认进行路由维护时，重传时间和重传尝试被调度的时间通常由网络中使用的特定链路层实现控制。例如，在IEEE 802.11中，作为IEEE 802.11分布式协调函数(DCF) MAC协议基本访问方法的一部分，在数据包后返回链路层确认;确认预期到达的时间和下一次重传尝试发生的时间由MAC协议实现控制。

当节点收到维护缓冲区中任何包的链路层ACK时，该节点应该从维护缓冲区中删除该包以及具有相同下一跳目的地的维护缓冲区中的任何其他包。

### 4.3.2 使用被动ACK

当链路层确认不可用，而被动确认可用时，在沿着最后一跳以外的任何一跳发起或转发包时，应使用被动确认来进行路由维护。特别是在这种情况下，如果节点可以将其网络接口置于“混杂”接收模式，且用于数据包的网络链路一般是双向运行的，则需要使用被动确认进行路由维护。

节点不能试图使用被动路由路线维护包的发起或转发的最后一跳(跳导致IP数据包的目的地址节点),因为接收结点不会转发数据包,因此没有这个节点可以听到的被动ACK。除了这个限制之外，节点还可以使用各种策略来使用被动确认来维护它发起或转发的包的路由。例如，以下两种策略是可能的:

1. 每当一个节点接收到一个要转发到最终目的地以外的节点的包时，该节点发送该包的原始传输，而无需请求网络层确认。如果在此传输之后，在PassiveAckTimeout内没有接收到被动确认，节点将重新传输数据包，同样不需要请求网络层确认;每次尝试使用相同的PassiveAckTimeout超时值。如果在数据包的TryPassiveAcks重传之后没有收到任何确认，则对该数据包的所有剩余尝试使用网络层确认(第3.3节所述)。

2. 每个节点维护一个可能的下一跳目标节点的表，指出通常是否可以期望从传输到该节点得到被动确认，以及来自该节点的被动确认的预期延迟和抖动。每当一个节点接收到要转发到IP目的地地址以外的节点的包时，该节点检查下一跳目的结点的表，以确定对该节点的传输是使用被动确认还是网络层确认。这个包的超时时间也可以从这个表推导出来。使用此方法的节点应该更喜欢使用被动确认，而不是网络层确认。

在对发出或转发的包使用被动确认时，如果以下两个测试都成功，则节点认为新包的后续接收就是对第一个包的确认:

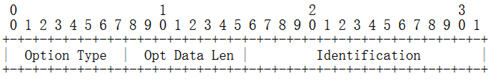
1.两个包的IP报头中的源地址、目标地址、协议、标识和片段偏移字段必须匹配[32]

2. 如果任何一个包包含DSR源路由报头，两个包都必须包含一个，并且新包的DSR源路由报头中DSR源路由报头左字段的段值必须小于第一个包中的值。

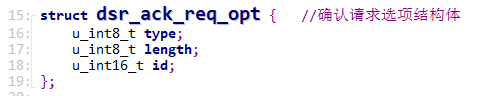
当一个节点从其维护缓冲区中的任何包听到这样的被动确认时，该节点应该从其维护缓冲区中删除该包以及具有相同下一跳目的地的维护缓冲区中的任何其他包。

### 4.3.3 使用网络层ACK

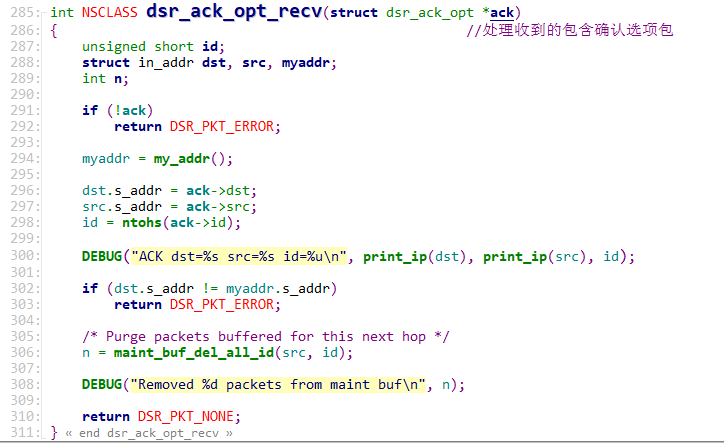
确认请求选项结构图：



源码如下：



18：id字段被设置为一个唯一的值，在接收到数据分组的节点返回时被复制到确认选项的id字段中。确认请求选项在DSR选项头中只能出现一次。



287-303：如果未确认，则返回错误，检查ack中的目的地址是否为自身地址，确认成功则将维护缓冲区中相应id的数据分组删除。

具体分析：

当一个节点发起或转发一个包，并且没有其他可用的确认机制来确定用于路由维护的源路由中的下一跳节点的可达性时，该节点应该从该下一跳节点请求一个网络层确认。为此，节点在包的DSR选项头中插入确认请求选项。该确认请求选项中的标识字段必须设置为该节点传输到同一下一跳节点的所有数据包的唯一值，这些数据包要么未被确认，要么最近被确认。

当节点接收到包含确认请求选项的包时，该节点对该包执行以下操作:

1.如果此包指定的下一跳节点地址与该节点自身的任何IP地址不匹配，则该节点必须不处理ACK请求选项。被指定的下一跳节点地址[i]字段在DSR选项头包中DSR源路由选项,或者是IP数据包的目的地址，如果包不包含一个安全域源路由选项或段左段是零。

2.如果包包含确认选项，则此节点必须不处理确认请求选项。

如果上述测试均未失败，则该节点必须通过向前一跳节点发送确认选项来处理确认请求选项;为此，节点执行以下步骤:

1. 创建一个包并将IP协议字段设置为分配给DSR的协议号

2. 复制自该包中的DSR源路由选项中的源路由，将此包中的IP源地址字段设置为该节点的IP地址。

3. 复制该包中的DSR源路由选项中的源路由，将此包中的IP目标地址字段设置为上一跳节点的IP地址。

4. 将DSR选项头添加到包中，并将DSR选项头的下一个标题字段设置为“No Next header”值。

5. 在包中的DSR选项头添加确认选项;将确认选项的选项类型字段设置为6,Opt Data Len字段设置为10。

6. 将接收到的确认请求选项中的标识字段复制到确认选项中的标识字段中。

7. 将确认选项中的ACK源地址字段设置为这个新包的IP源地址

8. 将确认选项中的ACK目的地地址字段设置为这个新包的IP目的地地址(上面设置为上一跳节点的IP地址)。

9. 按照1.1节讲述地那样发送包

包含确认选项的包不应放在维护缓冲区中。

当节点接收到同时具有确认选项和确认请求选项的包时，如果该节点不是确认选项的目的地(包的IP目的地地址)，则必须忽略确认请求选项。如果该节点是确认选项的目的地，该节点必须按照以下步骤顺序返回确认选项，以处理确认请求选项:

1. 1. 创建一个包并将IP协议字段设置为分配给DSR的协议号

2. 复制自该包中的DSR源路由选项中的源路由，将此包中的IP源地址字段设置为该节点的IP地址。

3. 将此包中的IP目标地址字段设置为发出确认选项的节点的IP地址。

4. 将DSR选项头添加到包中，并将DSR选项头的下一个头字段设置为“No Next header”值。

5. 在该包的DSR选项头中添加确认选项;将确认选项的选项类型字段设置为6,Opt Data Len字段设置为10。

6. 将接收到的确认请求选项中的标识字段复制到确认选项中的标识字段中。

7. 将选项中的ACK源地址字段设置为这个新包的IP源地址(上面设置为这个节点的IP地址)。

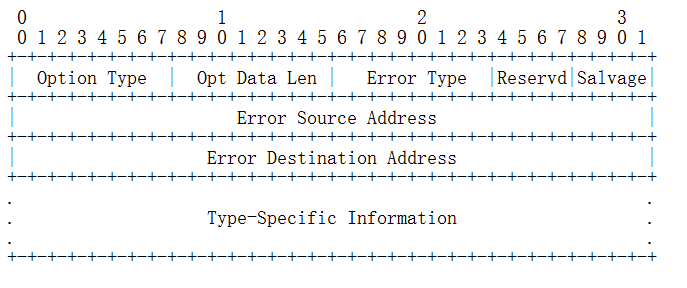
8. 将选项中的ACK目标地址字段设置为这个新包的IP目标地址(上面设置为发起确认选项的节点的IP地址)。

9. 把包直接送到目的地。IP目的地地址必须被视为一个直接的邻居节点:到该节点的传输必须在单个IP转发跃点中完成，不需要路由发现，也不需要搜索路由缓存。此外，此包不能包含DSR确认请求，不能为路由维护而重传，也不能期望链路层确认或被动确认。

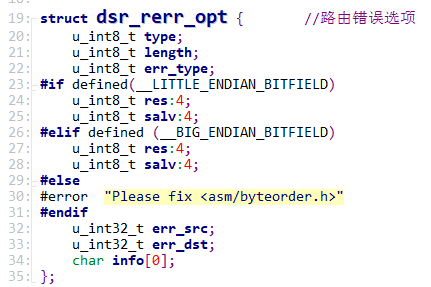
在使用网络层ACK进行路由维护时，节点应使用自适应算法确定请求每次传输尝试的重传超时。例如，对于最近试图传输数据包到的每一个节点，节点应该维护一个单独的往返时间(RTT)估计值，并且应该使用这个RTT估计值来设置每次路由维护重传尝试的超时。TCP RTT估计算法在DSR的实现和试验台实验中表现良好。

### 4.3.4 发起路由错误

路由错误选项结构图：



源码如下：

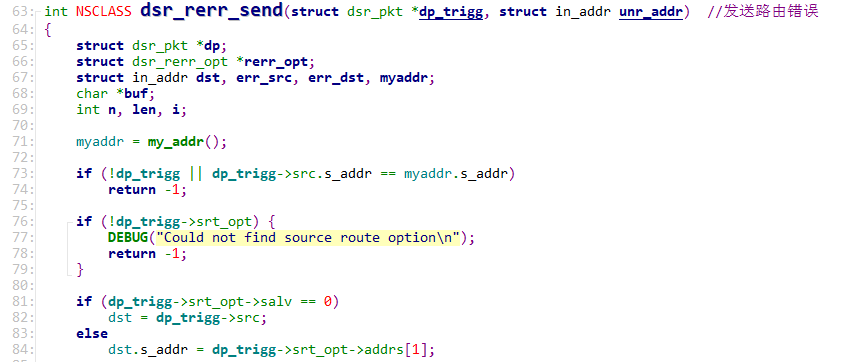


C:\Users\lenovo\AppData\Roaming\Tencent\Users\531973083\QQ\WinTemp\RichOle\3T4])8SFC0P6E[BG4JSCHLI.png

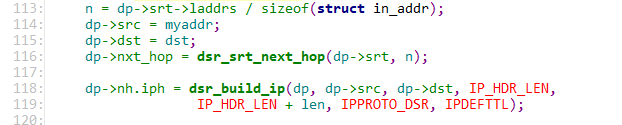
22-29：err\_type字段标记错误类型（其定义如图标号），salv从触发错误的数据包的源路由选项中的Salvage字段复制而来，标记数据分组被抢救的次数，避免数据包被无限抢救。

32-34：err\_src记录发起路由错误的节点地址，err\_dst记录错误目标地址，例如，当错误类型字段为NODE\_UNREACHABLE时，err\_dst会被设置为生成该路由信息的节点地址（即该节点认为此路径可达但事实上不可达）。Info数组记录有关此错误的信息。

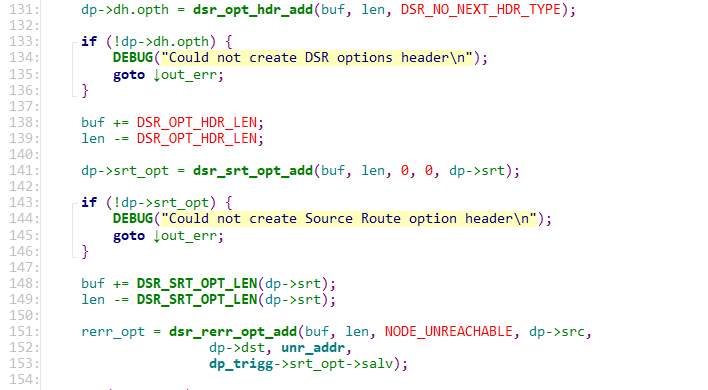
当一个节点在达到最大重发次数后依然不能验证下一跳节点的可达性时，它应该向该分组的IP源地址发送路由错误，代码实现如下：



C:\Users\lenovo\AppData\Roaming\Tencent\Users\531973083\QQ\WinTemp\RichOle\462~ZHZCE[ZETT1A[{J$)%X.png



65-119：新建一个数据包，将源地址设置为节点自身地址。如果触发路由错误的数据分组中抢救次数为0，则将目的地址设置为该触发此路由错误的数据分组中的源节点地址，否则将目的地址设置为addrs数组中的addr[1]。随后调用dsr\_build\_ip装填IP报头相关字段。其中93行在自身节点路由缓存中查找通往目的节点的路由，便于后续发送路由错误。



131-154：随后添加DSR路由选项头，向选项头中添加源路由选项，携带之前查找到的路由信息。同时向选项头中添加路由错误选项，类型设置为NODE\_UNREACHABLE，不可达地址设置为源路由中下一跳地址。

同时如果触发路由错误的数据包包含任何路由错误或确认选项，则节点可以将这些选项中的每一个追加到其路由错误中。后续代码中使用for循环实现追加选项。

具体分析：

当节点达到最大重传次数后无法验证下一跳节点的可达性时，节点应将路由错误发送到包的IP源地址。

当为包含路由错误选项或确认选项的包发送路由错误时，节点应将这些现有选项添加到其路由错误中，但受以下描述的限制。

发送路由错误的节点必须执行以下步骤:

1. 创建一个IP包，并将该包的IP报头中的源地址字段设置为该节点的地址。

2. 如果触发路由错误的包的DSR源路由选项中的打捞字段为零，则将触发路由错误的包的源地址字段复制到新包的IP报头的目的地址字段中;否则，将触发路由错误的包的DSR源路由选项中的地址[1]字段复制到新包的IP报头中的目标地址字段中。

3. 在新包中插入一个DSR选项头。

4. 路线错误选项添加到新包,设置错误类型NODE\_UNREACHABLE,残值到残值来自DSR数据包的源路由选项触发的路线错误,不可到达的节点地址字段到下一跳结点的地址来自于源路由。将错误源地址字段设置为该节点的IP地址，将错误目标字段设置为新包的IP目的地地址。

5. 如果触发路由错误的包包含任何路由错误或确认选项，节点可以将这些选项中的每一个附加到其路由错误上，其约束条件如下:

a. 节点不能包含触发新路由错误的包中的任何路由错误选项，因为其中包含的路由错误的总回收计数将大于新包中的MAX\_SALVAGE\_COUNT。

b. 如果包中不包含触发新路由错误的包中的任何路由错误选项，则节点必须不包含触发新路由错误的包中的任何后续路由错误或确认选项。

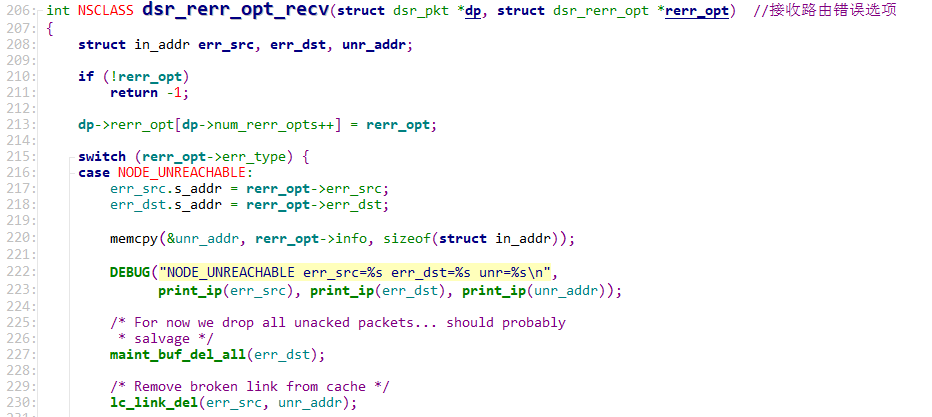
c. 触发路由错误的包中的任何附加选项必须遵循包中的新路由错误。

d. 在将这些选项附加到新路由错误时，必须保留触发路由错误的包中这些选项的顺序。

之后按照1.1节所讲述的发送包。

### 4.3.5 处理接收到的路由错误选项

当节点收到一个包含路由错误选项的数据包时，该节点从其路由缓存中删除从由错误源地址节点到由不可达节点（如果该路由存在于其路由缓存中）的路径信息。:



206-230：如果错误类型为不可达，则输出不可达地址信息，并将维护缓冲区中所有等待目标地址返回ack确认的数据包删除，同时删除不可达路径，更新自己的路由缓存。

所有的路由选项在函数dsr\_opt\_recv中处理，在while循环中使用switch语句对不同的选项调用相关的处理函数进行处理，所以所有附加的错误选项在213行通过自增操作都可以被处理。

具体分析：

当一个结点接收到了一个包含路由错误选项的包时，该结点必须按照下列步骤处理该理由错误选项：

1. 节点必须从其路由缓存中删除从错误源地址字段标识的节点到无法到达的节点地址字段标识的节点的连接。如果节点将其路由缓存实现为链接缓存，则只删除该单一链接;但是，如果节点将其路由缓存实现为路径缓存，则使用该链接的所有路由(路径)都将被删除。

2. 如果跟随路线错误的选项是一个由该节点发送确认或路线错误选项，将当前路由错误之后的DSR选项复制到一个新的包中，并使该包的IP源地址等于该节点自己的IP地址，IP目标地址等于ACK或错误目标地址。DSR源路由选项中的残值计数设置为路由错误的残值。

另外，此外,按上述处理路由错误之后,节点可能为任意一个目的结点启动一个新的路由发现。例如，如果该节点与某个目标节点有一个开放的TCP连接，那么如果该路由错误的处理从该节点的路由缓存中删除了到该目标的唯一路由，那么该节点可能会为该目标节点发起一个新的路由发现。但是，任何节点必须限制它为任何单个目的地地址发起新路由发现的速度，并且以这种方式发起的任何新路由发现(作为处理此路由错误的一部分)必须符合此限制。

### 4.3.6 补救一个包

如果一个中间结点转发一个包过程中通过路由维护发现该包在该路由上的下一跳的链路缺失，并且该结点在其路由缓存中有另一条到达该包ip目的地址的路由，节点应“补救”包而不是丢弃它。为了使用在其路由缓存中找到的路由，该节点对数据包的处理如下:

1.如果网络中使用的MAC协议不能通过单向链路传输单播数据包，如3.3.1节所述，则如果该数据包包含路由应答选项，则删除并丢弃该数据包中的路由应答选项;如果包中的DSR选项头不包含DSR选项，则从包中删除DSR选项头。如果生成的包只包含一个IP报头，则节点不应该回收包，而应该丢弃整个包。

2.修改包中现有的DSR源路由选项，使Address[i]字段表示在该节点的路由缓存中找到的到该包的IP目标地址的源路由

3.初始化DSR源路由选项中的段左字段为n。

4.DSR源路由选项中的(F)位是从标记包源路由中第一跳的外部位复制的，如路由缓存中所示。

5.DSR源路由选项中的(L)位是从标记包源路由中最后一跳的外部位复制的，如路由缓存中所示。

6.DSR源路由选项中的打捞字段设置为1并且加上引起错误的数据包的DSR源路由选项中的打捞字段的值。

7.使用第1.5节中描述的转发过程，将包发送到包中新的源路由上的下一跳节点。

如第3.4节所述，这种情况下的节点还应该向包的原始发送方返回一个路由错误。节点应该在引发路由错误后选择补救包。

# 第五章 总结

经过长达6周的文档阅读与代码分析，我们组成员将研究结果几乎全都写入文档中，在最后，我们给出了DSR路由协议的优缺点：

优点:

采用源路由机制、避免了路由环路。

它是一种按需路由协议、只有当两个节点间进行通信时，才会缓存路由纪录，因此相对主动路由来说，减小了路由维护的开销。

通过采用路由缓存技术，减少路由请求信息对信道的占用。

缺点:

随着路经跳数的增加，分组头长度线性增加、开销大。

路由请求分组RREQ采用洪泛发向全网扩散，导致网络负荷大。

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。

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。解决办法：若某节点听到其它节点发出的RREQ分组中路由信息含有较少跳数，此节点停止发送。

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，但有些路由信息可能是过时的。解决办法：引入定时器、链路断的情况应进行全网洪泛。