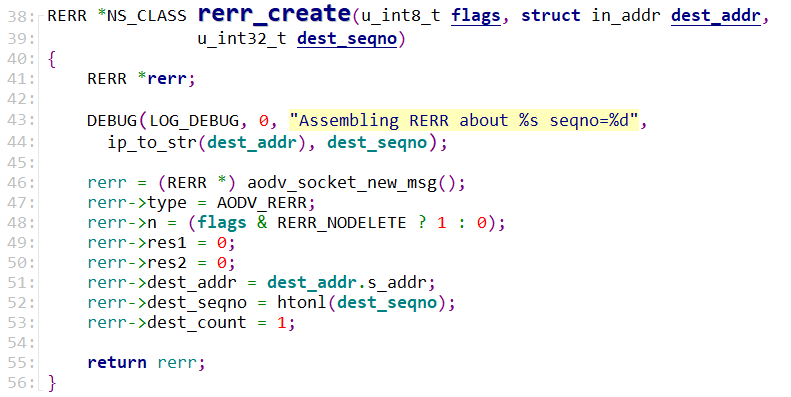
# 总览

|  |  |  |
| --- | --- | --- |
| 函数名 | 返回值 | 功能 |
| NS\_CLASS rerr\_create(u\_int8\_t flags, struct in\_addr dest\_addr,u\_int32\_t dest\_seqno) | RERR \* | 创建一条rerr消息 |
| NS\_CLASS rerr\_add\_udest(RERR \* rerr, struct in\_addr udest,u\_int32\_t udest\_seqno) | void | 定义一个RERR\_udest\*类型的变量ud来储存不可达节点的信息，rerr错误信息被更新 |
| NS\_CLASS rerr\_process(RERR \* rerr, int rerrlen, struct in\_addr ip\_src,struct in\_addr ip\_dst) | void | 从rerr消息中得到不可达目的节点，将通往不可达节点的路由rt置为无效路由并最后删除，同时向自己的邻居转播RERR消息 |

# 分析

## RERR \*NS\_CLASS rerr\_create(u\_int8\_t flags, struct in\_addr dest\_addr, u\_int32\_t dest\_seqno)



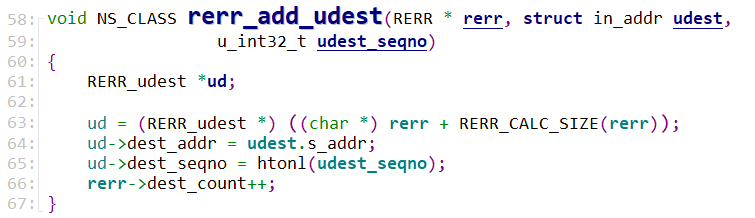
41 创建rerr数据包

43~44 设置输出信息

46 为新产生的数据包分配空间

47~53 给各项参数赋值

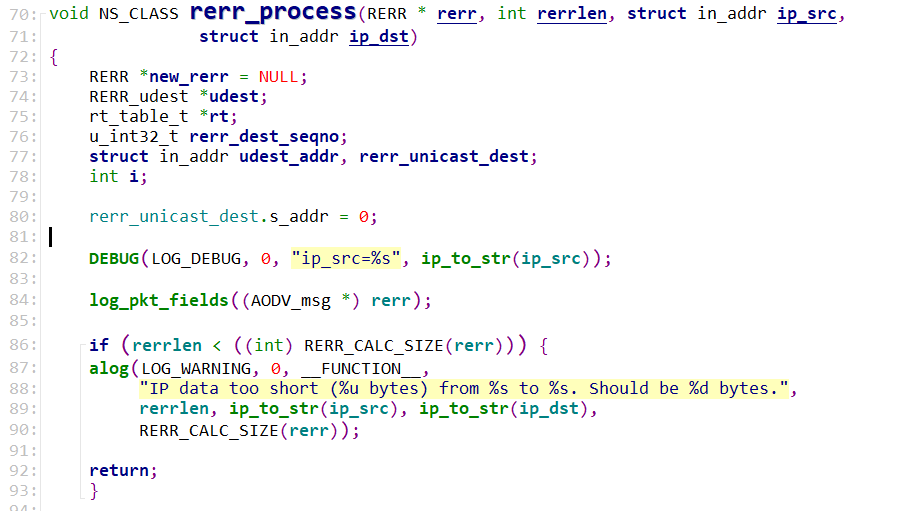
## void NS\_CLASS rerr\_add\_udest(RERR \* rerr, struct in\_addr udest,u\_int32\_t udest\_seqno)



61~63 定义一个RERR**\_**udest\*类型的变量ud来储存不可达节点的信息

64~66 ud记录不可达节点的IP地址及序列号等信息，rerr错误信息被更新

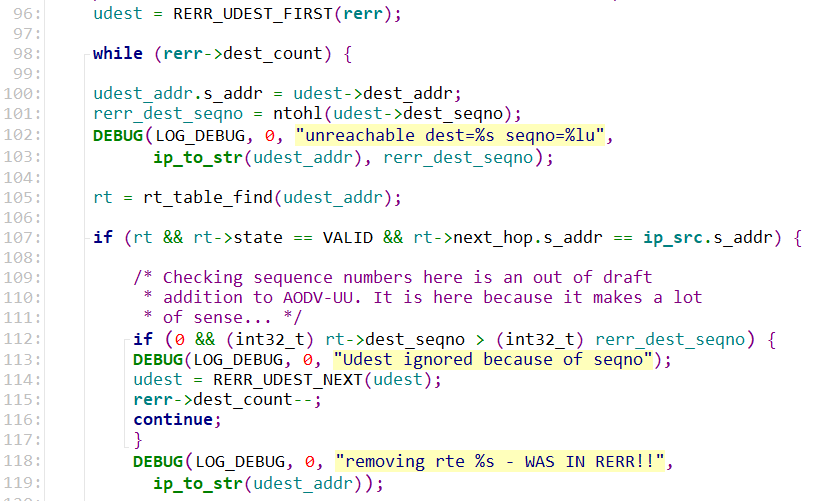
## void NS\_CLASS rerr\_process(RERR \* rerr, int rerrlen, struct in\_addr ip\_src,struct in\_addr ip\_dst)



73~80 一些变量的声明、初始化

82 打印调试信息

84 写入日志信息



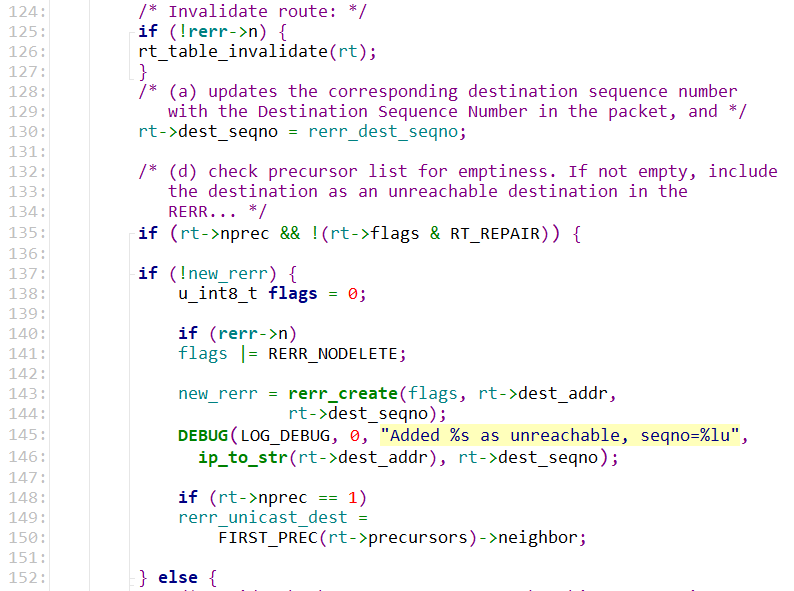
86~92 确定接收到的rerr消息的大小，如果其长度小于下限值则输出错误信息

96~103 从rerr消息中得到不可达目的节点，并对函数中定义的变量赋值，输出调试信息为不可达目的节点的IP地址和序列号

105 寻找到达不可达节点的路由表项

107~117 如果该路由表项仍有效并且下一跳目的节点与发送rerr消息的节点相同则利用目的节点序列号的比较判断消息是否过期，如果是过期消息将忽略

118~119 打印调试信息



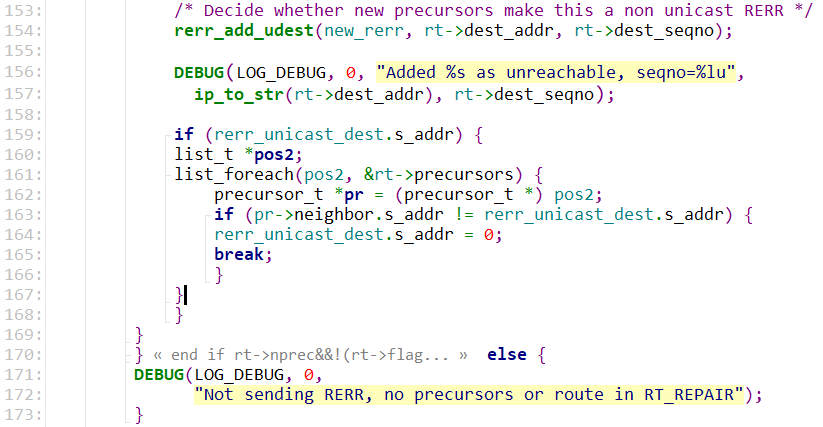
125~126 如果rrer消息未过期，则将通往不可达节点的路由rt置为无效路由

130 更新路由表项里的目的节点序列号

135 如果先驱表不空，路由修复也失败

137~146 新建一个rerr消息，对其各项参数赋值，并输出调试信息

148~150 如果只有一个需要接受rerr消息的邻居节点，则应向邻居节点单播

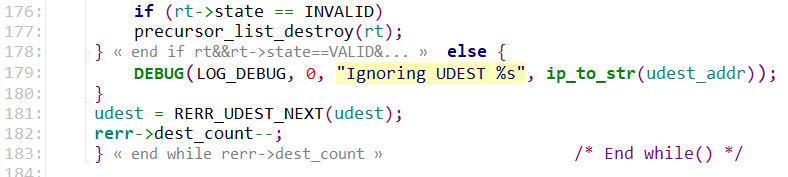


154 添加新的不可达节点

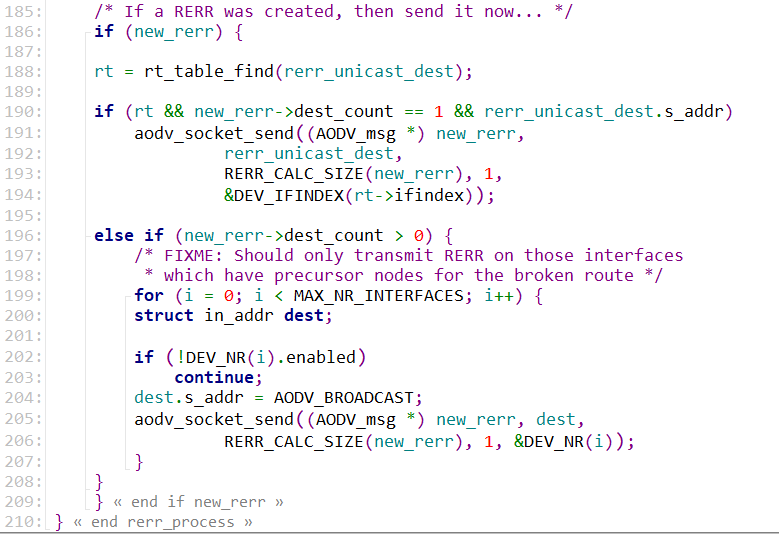
156~157 输出调试信息

159~168 是否有新的先驱节点使其成为非单播的rerr

171~172 输出调试信息



176~182 所有无效路由都将被删除并输出调试信息，处理下一个不可达节点



186~194 对于在路由表里能找到有效路由表项的节点，rerr消息直接发出

196~206 没有有效路由且其邻居节点大于0，则从所有可以发送消息的接口把rerr消息广播出去