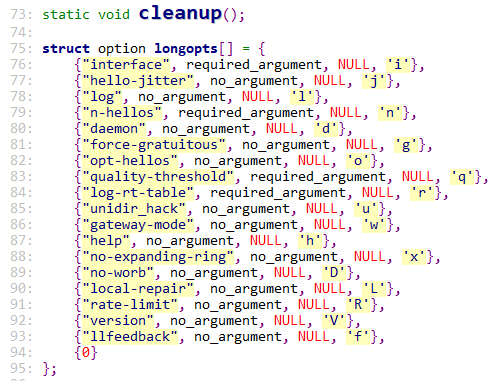
总览

|  |  |  |
| --- | --- | --- |
| 函数名 | 返回值 | 功能 |
| usage(int status) | Void | 如果状态值为0则输出usage信息，不为0打印标准错误提示用户进行帮助查看 |
| set\_kernel\_options() | int | 把ip\_forward写入on，置为开启。把所有接口关闭接口下发送和接受重定向的文件写入终止符off，在所有接口禁用ICMP重定向 |
| find\_default\_gw(void) | int | 寻找默认网关 |
| get\_if\_info(char \*ifname, int type) | struct sockaddr\_in \* | 获取指向名为ifname的接口地址的指针，如果设备驱动失败返回空指针 |
| attach\_callback\_func(int fd, callback\_func\_t func) | int | 增加一个文件描述符为fd，功能函数为func的回调结构元素，同时限制当前所拥有的套接字的处理函数和文件描述符数量不超过限制 |
| load\_modules(char \*ifname) | void | 根据接口的名字来获取其对应的设备信息以加载模块，没有则调用modprobe来加载模块。加载完成后还会检查是否加载完成。 |
| remove\_modules(void) | void | 调用system函数卸载模块 |
| host\_init(char \*ifname) | void | 主机初始化。如果没有接口，则先寻找无线接口。之后将rreq\_id初始化为0，已启用接口数为0，查找要在其上广播的所有接口并存储其信息，加载内核模块，启用IP转发并设置其他内核选项。 |
| signal\_handler(int type) | void | 信号处理器，使用switch结构根据信号的种类做不同处理。如果是SIGSEGV则写一条日志，其他信号不做处理。 |
| cleanup(void) | void | 静态函数，用于清空路由表、套接字、关闭日志、关闭套接字、卸载模块等。 |

## static void cleanup();



静态函数，指明了一个结构体option类型的长参数longopts的名称和性质。

76~94 {name,has\_arg,flag,val}

name:长参数名

has\_arg:为required argument表示该参数后一定要跟个参数值，为no\_argument表示该参数后不跟参数值

flag:为NULL表示getopt\_long()返回最后一个参数val

val:发现长选项时的返回值

## void usage(int status)

如果状态值为0则输出usage信息，不为0打印标准错误提示用户进行帮助查看

## int set\_kernel\_options()

把ip\_forward写入on，置为开启。把所有接口关闭接口下发送和接受重定向的文件写入终止符off，在所有接口禁用ICMP重定向

## int find\_default\_gw(void)

寻找默认网关

## struct sockaddr\_in \*get\_if\_info(char \*ifname, int type)

获取指向名为ifname的接口地址的指针，如果设备驱动失败返回空指针

## int attach\_callback\_func(int fd, callback\_func\_t func)

增加一个文件描述符为fd，功能函数为func的回调结构元素，同时限制当前所拥有的套接字的处理函数和文件描述符数量不超过限制

## void load\_modules(char \*ifname)

根据接口的名字来获取其对应的设备信息以加载模块，没有则调用modprobe来加载模块。加载完成后还会检查是否加载完成。

## void remove\_modules(void)

调用system函数卸载模块

## void host\_init(char \*ifname)

主机初始化。如果没有接口，则先寻找无线接口。之后将rreq\_id初始化为0，已启用接口数为0，查找要在其上广播的所有接口并存储其信息，加载内核模块，启用IP转发并设置其他内核选项。

## void signal\_handler(int type)

信号处理器，使用switch结构根据信号的种类做不同处理。如果是SIGSEGV则写一条日志，其他信号不做处理。

## int main(int argc, char \*\*argv)



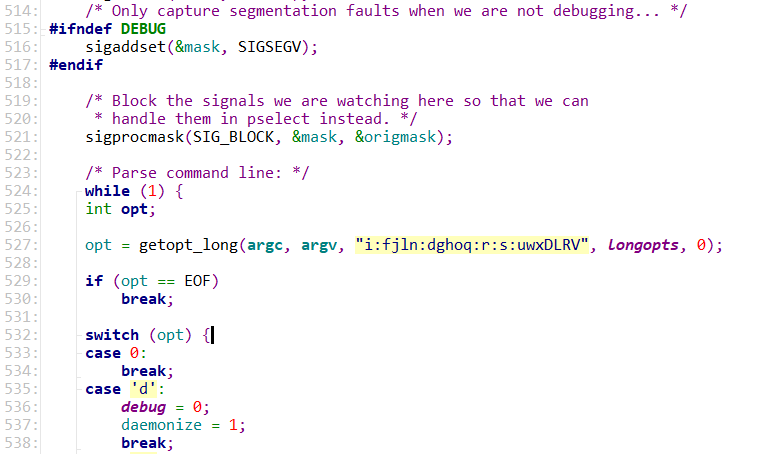
482~489 声明一些变量

492~498 设置程序名为其地址

501 打开调试输出开关

507~509 服务器关闭一些信号

511~513 把这些信号放进屏蔽信号集里



515~517 如果调试开启则屏蔽SIGSEGV信号

521 阻塞正在观看的信号之后可以使用pselect来处理

523~… 处理命令行操作，使用循环结构，根据用户输入进行不同操作，其各个输入参数在usage函数中有定义。

//等待翻译

"-d, --daemon Daemon mode, i.e. detach from the console.\n"

"-g, --force-gratuitous Force the gratuitous flag to be set on all RREQ's.\n"

"-h, --help This information.\n"

"-i, --interface Network interfaces to attach to. Defaults to first\n"

" wireless interface.\n"

"-j, --hello-jitter Toggle hello jittering (default ON).\n"

"-l, --log Log debug output to %s.\n"

"-o, --opt-hellos Send HELLOs only when forwarding data (experimental).\n"

"-r, --log-rt-table Log routing table to %s every N secs.\n"

"-n, --n-hellos Receive N hellos from host before treating as neighbor.\n"

"-u, --unidir-hack Detect and avoid unidirectional links (experimental).\n"

"-w, --gateway-mode Enable experimental Internet gateway support.\n"

"-x, --no-expanding-ring Disable expanding ring search for RREQs.\n"

"-D, --no-worb Disable 15 seconds wait on reboot delay.\n"

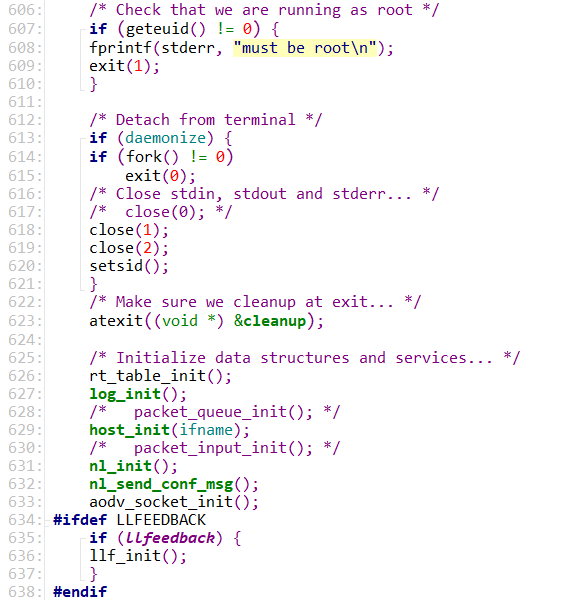
"-L, --local-repair Enable local repair.\n"

"-f, --llfeedback Enable link layer feedback.\n"

"-R, --rate-limit Toggle rate limiting of RREQs and RERRs (default ON).\n"

"-q, --quality-threshold Set a minimum signal quality threshold for control packets.\n"

"-V, --version Show version.\n\n"

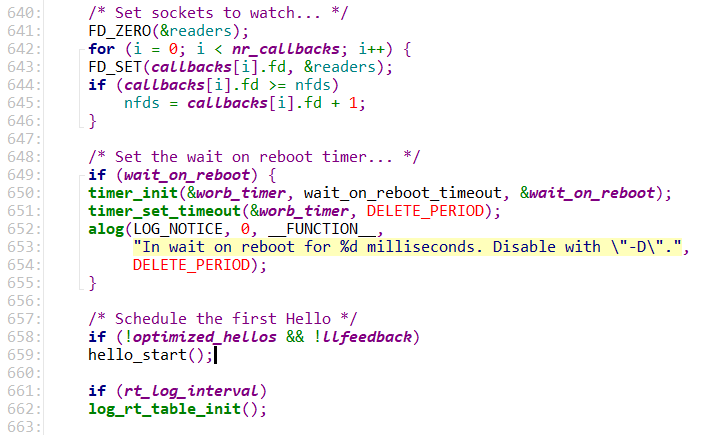


607~610 检查确定自己是在root权限下

612~621 如果daemonize 变量不为0，则关闭标准输入输出流和标准错误流，当前进程如果是父进程则直接退出，如果是子进程则并且使当前进程成为新的会话的领头进程，并与其父进程的会话组和进程组脱离。由于会话对控制终端的独占性，进程同时与控制终端脱离。

623 atexit函数是正常程序退出时调用的函数，我们把他叫为登记函数

626~637 各个数据结构和服务的初始化



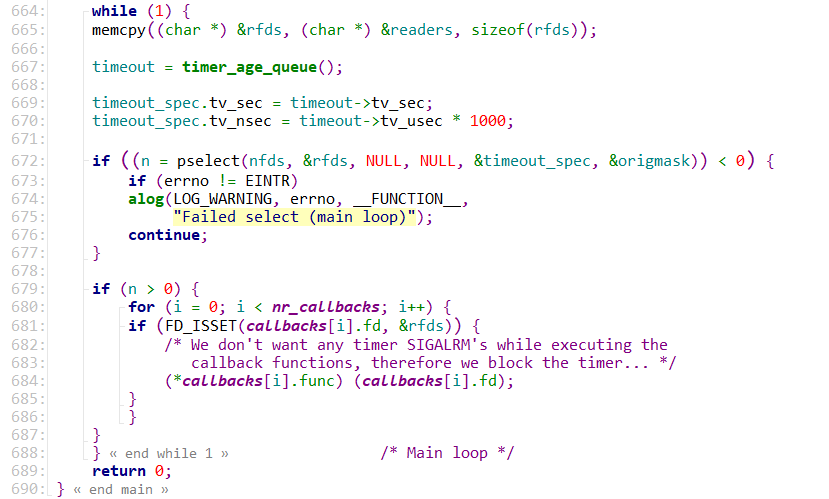
641 将指定的文件描述符清空

642~646 添加进文件描述符集合，更新文件描述符数量

648~655 设置超时重启定时器并写日志

657~659 准备第一个hello帧

661~662 初始化路由表



一个while永真循环

665 把文件描述符readers拷贝到rfds指向的数组中。

667~670 设置超时定时器

672~677 使用pselect函数进行IO复用，监视多个文件描述符的集合，用超时轮询的方式查看文件的读写错误可操作性，判断是否有符合条件的事件发生。

679~687 如果有一个或多个事件发生则查看其文件描述符如果在集合里则调用相应的回调函数来处理。

## static void cleanup(void)

清空路由表、套接字、关闭日志、关闭套接字、卸载模块等。