Posts - 31, Articles - 0, Comments - 19 | Cnblogs | Dashboard | Login |

代码改变世界

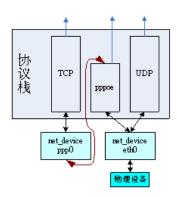
zmkeil

HOME CONTACT GALLERY

PPP协议体系的实现

2013-05-02 22:05 by zmkeil, 1830 阅读, 0 评论, 收藏, 编辑

其实PPP不像是一种协议,而更像是一种应用,可以把它看成一个拨号上网的应用软件,拨号成功后,本地主机就可以正常上网了,可以使用TCP/IP等协议,而完全感觉不到PPP的存在。而实际上PPP在网络协议栈中增加了不少东西,但对上层透明。另外PPP一般需要底层工具来支持,如之前讲的PPPoE。

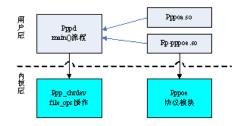


Pppoe协议的实现在协议栈中,且其底层有实际的物理设备(或者vlan设备)支持,关键就在于pppoe协议可以直通应用层(如上一篇所讲),也可以半途连接pppo设备。而ppp协议的实现主要在设备pppo的驱动中,这是一个虚拟设备,没有物理设备的支持,且保持对上层是透明。

1.代码概述

Ppp是一个应用程序,其代码在ppp软件包中,其中主体部分最终编译成pppd文件,另外还有很多支持部件(plugin),如pppoe、pppoa等,在子文件夹plugin中,最终编译成rp-pppoe.so、pppoa.so文件。

在内核中,对此的支持主要也有两部分,一个是ppp模块(注意这不是最终的ppp网络设备,而是一个字符设备,主要对构建维护ppp连接提供支持),另一部份是pppoe协议模块(其实还包括裸packet协议模块),如下图左所示:



最新评论

Re:Luci实现框架

您好,想请教一个问题,我想将Luci的admin-full下面的syslog显示功能移植到admin-mini,请问怎么实现? -- zyzferrari

| | | | 日历 | | | | 随笔档案 |
|-------------|----|-----------|----------|----|----|-----------|-------------|
| < 2013年5月 > | | | | | | | 2016年5月(2) |
| Ħ | _ | \equiv | \equiv | 四 | 五 | 六 | 2016年2月(1) |
| 28 | 29 | 30 | 1 | 2 | 3 | 4 | 2015年11月(1) |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 | 2015年2月(1) |
| 12 | 13 | <u>14</u> | 15 | 16 | 17 | 18 | 2015年1月(1) |
| 19 | 20 | <u>21</u> | 22 | 23 | 24 | <u>25</u> | 2013年8月(3) |
| <u>26</u> | 27 | 28 | 29 | 30 | 31 | 1 | 2013年5月(9) |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 2013年4月(13) |
| | | | | | | | |

随笔分类

| Linux开发杂记(4) |
|-------------------|
| 编程语言C/C++/JAVA(5) |
| 操作系统(4) |
| 计算机架构(1) |
| 算法(2) |
| 网络相关(15) |
| 信号处理DSP(2) |
| 有感而发(4) |
| |

推荐排行榜

- 1. Linux下的虚拟Bridge实现(4)
- 2. 网络嵌入式设备(2)
- 3. 关于uC/OS的简单学习(2)
- 4. Luci实现框架(2)
- 5. uhttpd的实现框架(2)

阅读排行榜

- 1. Luci实现框架(12752)
- 2. uhttpd的实现框架(4069)

```
Int main(argc,argv)
                                    //# Fiphigin
   option_from_arg(argc,argv)
   for(;;) {
     \underline{\log} \operatorname{open}(0)
                        //自动ppp连接
     start_link(0)
     whilephase != DEAD) {
                                    #字体部分
         handle_events()
         getimput()
                        //处理ppp事务
                //真正的数据通信在hannel中
     reagain the channel //处理ppp 意外中断
                        /域退出
     or exit
```

3. Linux下的虚拟Bridge实现(3963)

4. OpenWRT平台搭建及简单应用 (3162)

5. Linux下VLAN功能的实现(1967)

如上图右所示,是pppd程序的main流程,首先是从argv(或者从file中)中获得plugin信息,并加载;然后执行一个for循环,主要是为了处理ppp连接意外中断、或设备暂时中断的情况下,可以重新启动连接;与ppp协议相关的部分是通过start_link(o)启动一个ppp连接,然后执行while循环,处理各种ppp信息;最后要说明的是,真正的数据通信是在建立的ppp连接中进行的,与这里的pppd没有关系,而ppp连接中的信息却可以发送到这里的while循环中来,具体实现后面会描述。

由于整个代码比较复杂,没有去详细看,只就其中一些关键点做一些了解。

2. plugin机制实现

Ppp有很多种类型的plugin,如pppoe、pppoa等,这些plugin最终被编译成多个xx.so文件,并提供一个通用的struct channel{}接口,如下图左所示。一般的pppd命令如下:

Pppd plugin rp-pppoe.so etho

和常见的命令结构一样,由option+arg的方式构成,上述的命令中体现了两个选项,一个是plugin,其参数为rp-pppoe.so,另一个是etho,没有参数。

在pppd程序中,选项option由一个特殊的结构来描述,如下图右所示:

Struct channel
options
process_extra_option
check_options
connect
disconnect
establish_ppp
disestablish_ppp

Struct option_t
cher*name
opt_type type
cher*description
int flags
void*addr2

上述的两个选项对应的option t结构分别为:

{"plugin",o_special,(void*)loadplugin,"load a plugin module into pppd",......}

{"device-name",o_wild,(void*)&PPPoEDevnameHook,"pppoe device name".....}

在option_from_args()函数中,主要通过parse_option()函数来处理args选项参数,其中首先利用find_option()函数来识别选项,并找到对应的option_t结构,一般采用name匹配方法,如"plugin",有些则特殊,如"devicename"。然后调用process_option()函数来处理。

```
option_from_args(argc,argv)
                                                     ocess_option(opt,arg,argv)
                                                      switck(opt->type)
Parse_args(argc,argv)
                                                      case o special:
   while arge>0) {
                                                         parser= (int(*)(char **))opt->addr2
      arg= *argv+-
                                                         (*parser)(arg)
                                                         break
       opt= find_option(argv)
                                                     case o_wild:
       process_option(opt,arg,argv)
                                                         wildp= (irt(*)(char*,char**,irt))opt->addr2
                                                         (Wildp)(argargv,1)
       argv+= n
                                                         break
```

Plugin选项对应的处理函数为loadplugin(),该函数利用标准库函数 dlopen(),打开plugin文件rp-pppoe.so,然后再利用标准库函数dlsym(),找到其中的plugin_init()函数。这种方法适用于这样的情况:由多种动态连接库供应用程序选择,且每个动态库(.so)中都定义了一个名称相同的函数。Plugin_init()函数的处理很简单,就是把自身库特有的选项加入到全局选项表中去,上述的第二个选项{"device-name",......"pppoe device name"}就是此时加入的,当然不同的动态库中的plugin_init()函数中,加入的选项各不相同。

这样就好理解第二个选项了,其处理函数是rp-pppoe.so中的私有函数 PPPoEDevnameHook(arg,argv,1),该函数是加载pppoe-plugin的关键:

```
Int PPPoEDevnameHook(cml, argv, l)
fd= socke(PF_PACKETSOCK_RAM_0)
struct ifteq iff ifr.ifr.name = cmd
ioct(fd,SIOCGIFINDEX_&iff)
ioct(fd,SIOCGIFINDEX_&iff)
ioct(fd,SIOCGIFINDEX_&iff)
detective ifinfr_lawaddr.sa_family = ETHERNET //判断interface是否为以太网
//-下设置全局变量
stmcpf,devnam.cmd_len) //复制设备名称到evnam中
the_channel = &pppoe_channel //详全局变量the_channel 设置为pppoe_channel
modem= 0
PPPoEInitDevice() //初始化PPPoEComection.com变量
```

该函数首先利用socket的ioctl函数,判断所给interface是否为以太网接口;然后将设备名和pppoe_channel分别赋给pppd程序空间中的devname和the_channel;最后对PPPoE模块进行一些简单的初始化。

这样之后,pppd程序就可以利用pppoe模块了,其它模块也一样。对于pppd而言,它不关心各种模块实现的细节,而只是调用各模块提供的统一接口the channel。

3.创建ppp连接通道

前面讲了,各种plugin模块提供给pppd的调用接口都相同,只是实现细节不一样,下面主要以pppoe为例,描述如何创建一个ppp连接通道。回到第1节图中的main()函数流程,可以看到创建连接是通过调用lcp_open(o),start_link(o)来完成的。具体细节就不看了,大致的流程是:Lcp_open()会创建一个ppp_netdev,如pppo,而start_link()函数则会创建一个pppoe的连接,作为pppo设备的传输通道,如下图所示。

```
Void lcp_open()
make_ppp_uint()
.....

Int make_ppp_unut()
ppp_dev_fd= open( '/dev/ppp' ,O_RDWR)
ioct[ppp_dev_fd,PPPIOCNEWUNIT,...)
.....
```

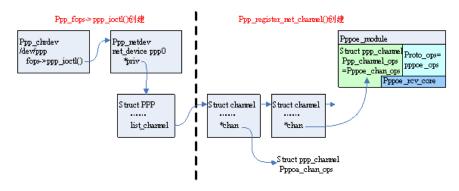
Void start_link(uint) devfd= the_channel->connect() fd_ppp = the_channel->establish_ppp(devfd)

"/dev/ppp"是一个字符设备ppp_chrdev,其对应的内核代码在ppp_generic.c中,对它的操作都通过标准的文件操作ppp_file_ops来调用,其中ioctl操作中,PPPIOCNEWUNIT选项对应于创建一个新的网络设备net_device(ppp_netdev),该设备的私有空间为一个struct ppp结构,而其网络操作由ppp_netdev_ops来调用,这些特有的结构和操作决定了ppp_netdev设备的功能与特性,后面还会详细描述。

Start_link()函数中,首先调用the_channel->connect()函数,对于PPPOE-

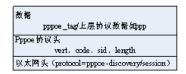
plugin来说,就是创建一个PF_PPPOE的socket接口,并调用该socket的标准connect操作,该操作在前一篇的第3节中已描述,除了处理协议地址外,还有一个关键操作就是ppp_register_net_channel(po->chan)。然后继续调用the_channel->establish_ppp(devfd),对于PPPOE-plugin,该函数为generic_establish_ppp(fd),它的操作很简单,但很关键,就是调用ioctl(fd,PPPIOCGCHAN,...),该操作在前一篇的第3节已描述,就是设置socket插口的state|=BOUND,使得它接收到的数据包都转交给ppp_netdev。

经过一些列创建操作后,系统模型如下图所示:

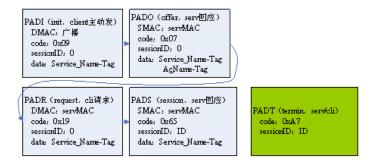


4.补充PPPOE协议本身

对于一个协议,其最具代表性的就是它的帧结构,pppoe的帧结构如下图所示:



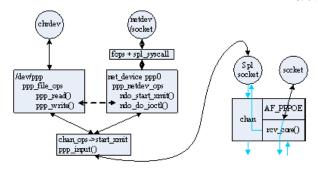
联系connect调用,需要用户提供pppoeaddr(其中包括dev、sessionID、remoteMAC),才能创建一个可通信的socket连接,我们称为pppoe的session连接。而怎么获得pppoeaddr呢?这其实是pppoe的发现阶段,其实现一般采用PF_PACKET协议,自己设置裸数据包为pppoe-discovery格式,具体流程如下图所示:



5.收发代码细节

PPP协议的的数据包结构如第4节图所示,其中pppoe的负载即为PPP协议 头+PPP数据,而PPP数据可以是一个完整的通用协议包,如IP包。本文开头 处的图很好地描述了PPP协议所需要做的工作,以及整个协议体系的构成、分 布情况。

首先给出整个收发流程的框架图,如下:

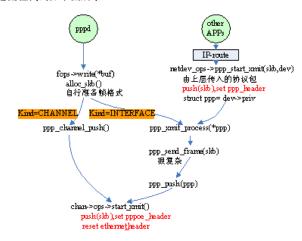


ppp模块可以通过网络接口或者文件接口来操作,它们最终都通过底层 pppoe_channel来完成数据通信,联系第一节中main()函数流程中的while循 环,它实际上就是通过文件接口,来监控ppp连接。

pppoe在协议栈中实现,但它用一个特殊的socket插口来服务于ppp,该 socket (即第3节所述的connect插口) 只对pppd程序可见,且从不直接用它 来发送数据,只是利用它注册的chan为ppp发送数据,而它收到的数据都通过 ppp_input()递交给ppp。这些并不会影响pppoe协议栈的工作,其他应用程序可以创建其它AF_PPPOE的socket插口,进行正常的pppoe通信。

5.1发送流程

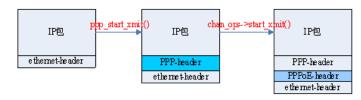
发送流程代码如下图所示:



pppd程序通过对/dev/ppp字符设备的操作,最终由ppp_chan实现与对端的数据通信,数据格式有ppp协议自己决定,主要用于维护ppp连接。

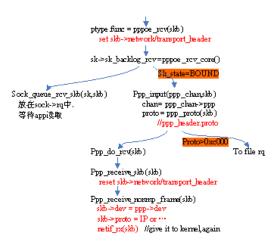
主要看一下ppp的网络设备,它有自己独立的net_device结构、IP地址等,对上层而言就像一个普通的设备,IP协议中的路由可以选择该设备。该设备的驱动程序是实现ppp协议的主要地方,它对skb进行push操作,添加ppp的协议头,最后调用pppoe_chan将数据包发送出去。注意pppoe_chan_ops和pppoe_ops没有太大区别,都要设置pppoe的协议头,只是pppoe_chan_ops接收到的skb是上层协议栈的,需要一个push操作,而pppoe_ops是自己分配skb。

数据包格式如下图所示:



5.2接收流程

接收流程代码如下图所示:



发送时逐层插入协议头,而接收时,一般不需要删除协议头(那样需要额外的put(skb)操作),而仅改变skb->network/transport_header即可。为了实现对上层的透明,接收流程的最后,重新设置dev为该pppo,proto为ppp负载数据的协议,然后调用netif_rx(skb)重新启动接收流程,这样该skb就仿佛是从pppo设备接收到的。

整个过程中,数据包几乎不变,仅是skb的相应字段改变,如上所述。

6.总结

仅对ppp应用的工作流程做了简单学习,了解了其实现的基本框架,对ppp 协议本身没有做深入学习。



刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论,请 登录 或 注册, 访问网站首页。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】融云即时通讯云一豆果美食、Faceu等亿级APP都在用

【推荐】报表开发有捷径: 快速设计轻松集成,数据可视化和交互

【推荐】一个月仅用630元赚取15000元,学会投资

【推荐】阿里舆情首次开放,69元限量秒杀



最新**IT**新闻:

- · 华为企业云发布一年考
- · 大老板的焦虑、寂寞和人才困境
- · 穷游网十二年, 一个老社区的演变和它的新生意
- ·微软推出Android测试版Flow自动化事务处理应用
- · IM企业热衷推出实体商品: Slack开售美式纹身贴纸
- » 更多新闻...



90%的开发者选择极光推送

不仅是集成简单、24小时一对一技术支持

最新知识库文章:

- · 程序猿媳妇儿注意事项
- ·可是姑娘,你为什么要编程呢?
- · 知其所以然(以算法学习为例)
- · 如何给变量取个简短且无歧义的名字
- · 编程的智慧
- » 更多知识库文章...