

u012845519的专栏

目录视图

摘要视图

RSS 订阅

个人资料



13459104362

访问: 2890次

积分: 43

等级: 

排名: 千里之外

原创: 1篇

转载: 0篇

译文: 0篇

评论: 4条

文章搜索

文章分类

mac80211 (1)

TrustZone (0)

TEE (0)

文章存档

2014年02月 (1)

阅读排行

mac80211源码分析

(2815)

评论排行

mac80211源码分析

(5)

推荐文章

*Android RocooFix 热修复框架

【公告】博客系统优化升级

【收藏】Html5 精品资源汇集

博乐招募开始啦

mac80211源码分析

2014-02-18 10:422819人阅读评论(5)收藏举报

分类: mac80211

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录(?)

[+]

mac80211源码分析

1. 概述

2. 体系结构

3. 代码结构

4. 数据结构

5. 主要流程

6. 切换点

7. 主要函数

8. 速率控制

1、概述

- mac80211: 是一个Linux内核子系统，是驱动开发者可用于为SoftMAC无线设备写驱动的框架。mac80211在内核空间实现STA模式，在用户空间实现AP模式（hostapd）。
- cfg80211: 用于对无线设备进行配置管理，与FullMAC，mac80211和nl80211一起工作。
- nl80211: 用于对无线设备进行配置管理，它是一个基本Netlink的用户态协议。
- MLME: 即MAC（Media Access Control）Layer Management Entity，它管理物理层MAC状态机。
- SoftMAC: 其MLME由软件实现，mac80211为SoftMAC实现提供了一个API。即：SoftMAC设备允许对硬件执行更好地控制，允许用软件实现对802.11的帧管理，包括解析和产生802.11无线帧。目前大多数802.11设备为SoftMAC，而FullMAC设备较少。
- FullMAC: 其MLME由硬件管理，当写FullMAC无线驱动时，不需要使用mac80211。
- wpa_supplicant: 是用户空间一个应用程序，主要发起MLME命令，然后处理相关结果。
- hostapd: 是用户空间一个应用程序，主要实现station接入认证管理。

2、体系结构

* android6.0源码分析之Camera API2.0下的初始化流程分析

*Android_GestureDetector手势滑动使用

*Android MaterialList源码解析

*Android官方开发文档Training系列课程中文版：创建自定义View之View的创建

最新评论

mac80211源码分析
eveduo: 请问 图2-1 系统框架 中 nl80211 和 cfg80211_ops 是不是位置反了？

mac80211源码分析
Y蓝羽Y: 感谢博主总结，学习了

mac80211源码分析
yutengao1987: 学习了 帧怎么抓呢

mac80211源码分析
13459104362: @u013785643: 开启monitor模式就可以抓到。

mac80211源码分析
lvy-cx: 学习了 博主，请问下在哪里能捕捉到控制帧和管理帧？

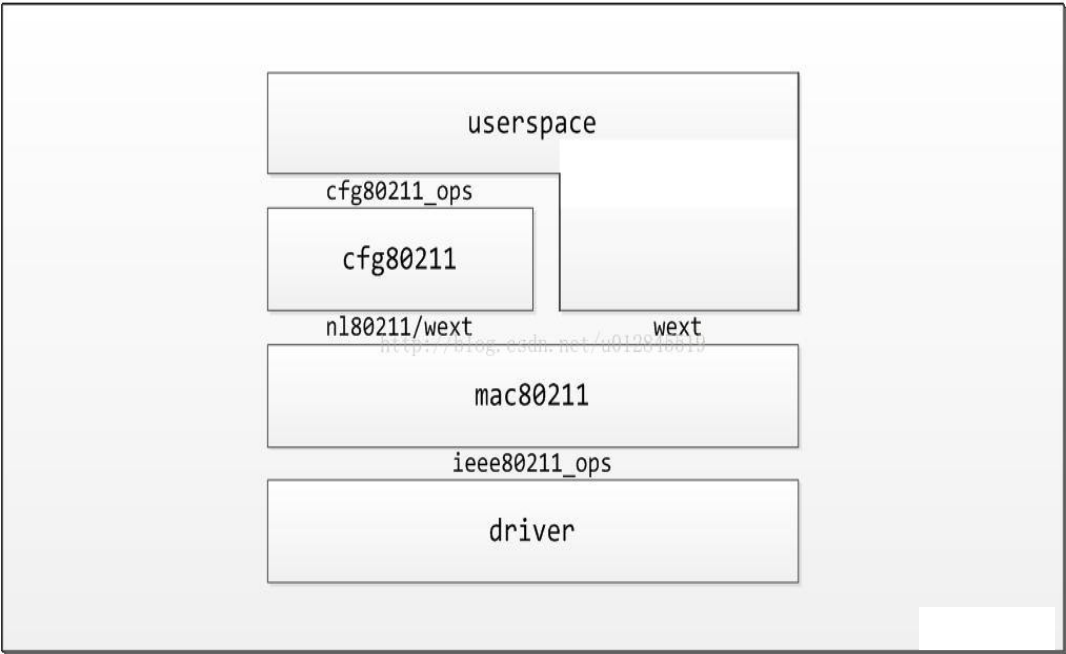


图2-1 系统框架

3、代码结构

- ieee80211_i.h（主要数据结构）
- main.c（主函数入口）
- iface.c（虚拟接口处理）
- key.c, key.h（密钥管理）
- sta_info.c, sta_info.h（用户管理）
- pm.c（功率管理）
- rate.c, rate.h（速率控制函数）
- rc80211*（速率控制算法）
- rx.c（帧接收路径代码）
- tx.c（帧发送路径代码）
- scan.c（软件扫描代码）
- mlme.c（station/managed模式MLME）
- ibss.c（IBSS MLME）
- cfg.c, cfg.h, wext.c（配置入口代码）
- aes*, tkip*, wep*, michael*, wpa*（WPA/RSN/WEP代码）
- wme.c, wme.h（QoS代码）
- util.c（公共函数）

4、数据结构

ieee80211_local/ieee80211_hw

- 每个数据结构代表一个无线设备（ieee80211_hw嵌入到ieee80211_local）
- ieee80211_hw是ieee80211_local在驱动中的可见部分
- 包含无线设备的所有操作信息

sta_info/ieee80211_sta

- 代表每一个station
- 可能是mesh，IBSS，AP，WDS
- ieee80211_sta是驱动可见部分

ieee80211_conf

- 硬件配置
- 当前信道是最重要的字段
- 硬件特殊参数

ieee80211_bss_conf

- BSS配置
- 多BSSes类型（IBSS/AP/managed）
- 包含比如基础速率位图
- per BSS parameters in case hardware supports creating/associating with multiple BSSes

ieee80211_key/ieee80211_key_conf

- 代表加密/解密密钥
- ieee80211_key_conf提供给驱动用于硬件加速
- ieee80211_key包含book-keeping和软件解密状态

ieee80211_tx_info

- 大部分复杂数据结构
- skb内部控制缓冲区(cb)
- 经历三个阶段：1、由mac80211初始化；2、由驱动使用；3、由发送状态通告使用

ieee80211_rx_status

- 包含接收帧状态
- 驱动通过接收帧传给mac80211

ieee80211_sub_if_data/ieee80211_vif

- 包含每个虚拟接口信息
- ieee80211_vif is passed to driver for those virtual interfaces the driver knows about
- 包含的sub-structures取决于模式

5、主要流程

配置

- 所有发起来自用户空间（wext或者nl80211）
- managed和IBSS模式：触发状态机（基于workqueue）
- 有些操作或多或少直接通过驱动传递（比如信道设置）

接收路径

- 通过函数ieee80211_rx()接收帧
- 调用ieee80211_rx_monitor()拷贝帧传递给所有监听接口
- 调用invoke_rx_handlers()处理帧
- 如果是数据帧，转换成802.3帧格式，传递给上层协议栈
- 如果是管理帧/控制帧，传递给MLME

接收处理钩子（invoke_rx_handlers）

- ieee80211_rx_h_passive_scan
- ieee80211_rx_h_check
- ieee80211_rx_h_decrypt
- ieee80211_rx_h_check_more_data
- ieee80211_rx_h_sta_process

- ieee80211_rx_h_defragment
- ieee80211_rx_h_ps_poll
- ieee80211_rx_h_michael_mic_verify
- ieee80211_rx_h_remove_qos_control
- ieee80211_rx_h_amsdu
- ieee80211_rx_h_mesh_fwding
- ieee80211_rx_h_data
- ieee80211_rx_h_ctrl
- ieee80211_rx_h_action
- ieee80211_rx_h_mgmt

发送路径

- 帧传递给ieee80211_subif_start_xmit()
- 把帧转换成802.11格式，丢弃发给未认证工作站的单播包，除了来自本地的EAPOL帧
- 如果是MONITOR接口，在帧头部增加radiotap信息
- 调用invoke_tx_handlers()处理帧
- 调用drv_tx()，把帧传递给驱动

发送处理钩子 (invoke_tx_handlers)

- ieee80211_tx_h_dynamic_ps
- ieee80211_tx_h_check_assoc
- ieee80211_tx_h_ps_buf
- ieee80211_tx_h_select_key
- ieee80211_tx_h_sta
- ieee80211_tx_h_rate_ctrl
- ieee80211_tx_h_michael_mic_add
- ieee80211_tx_h_sequence
- ieee80211_tx_h_fragment
- ieee80211_tx_h_stats
- ieee80211_tx_h_encrypt
- ieee80211_tx_h_calculate_duration

management/MLME

- 状态机运行依赖于用户请求
- 标准方法如下：
- probe request/response
- auth request/response
- assoc request/response
- notification request/response

IBSS

- 尝试寻找IBSS
- 加入IBSS或者创建IBSS
- 如果没有配对，则周期性地尝试寻找IBSS并加入

创建接口路径

- 创建接口由用户空间通过nl80211发起
- 分配网络设备空间（包含sdata对象空间）

- 初始化网络设备
- 初始化sdata对象（包括设备类型，接口类型，设备操作函数等等）
- 注册网络设备
- 把sdata对象加入local->interfaces

删除接口路径

- 删除接口由用户空间通过nl80211发起
- 把sdata对象从local->interfaces移除
- 移除网络设备

创建station路径

- 创建station由用户空间通过nl80211发起
- 分配sta_info对象空间
- 初始化sta_info对象（包括侦听间隔，支持速率集等等）
- 初始化sta_info对象的速率控制对象
- 把sta_info对象加入local->sta_pending_list
- 调用local->ops->sta_add通知驱动创建station
- 把sta_info对象加入local->sta_list

删除station路径

- 删除station由用户空间通过nl80211发起
- 删除sta_info对象的key对象
- 把sta_info对象从local->sta_pending_list移除
- 调用local->ops->sta_remove通知驱动移除station
- 删除sta_info对象的速率控制对象
- 把sta_info对象从local->sta_list移除

扫描请求路径

- 扫描请求由用户空间通过nl80211发起
- 如果支持硬件扫描，调用local->ops->hw_scan()执行硬件扫描
- 否则，调用ieee80211_start_sw_scan()执行软件扫描
- 延时唤醒ieee80211_scan_work()

扫描状态机路径

- 如果存在硬件扫描请求，调用drv_hw_scan()进行扫描，如果失败，调用ieee80211_scan_completed()完成扫描
- 如果存在扫描请求，同时未进行扫描，调用__ieee80211_start_scan()进行软件扫描，如果失败，调用ieee80211_scan_completed()完成扫描
- 根据next_scan_state调用相应的处理函数
- 如果next_delay==0，则继续根据next_scan_state调用相应的处理函数
- 延时唤醒ieee80211_scan_work()

6、切换点

配置

- wireless extensions (wext)
- cfg80211（通过nl80211和用户空间通信）

wext

- 设置SSID, BSSID和其他关联参数
- 设置RTS/fragmentation thresholds
- managed/IBSS模式的加密密钥

cfg80211

- 扫描
- 用户管理 (AP)
- mesh管理
- 虚拟接口管理
- AP模式加密密钥

从mac80211到速率控制

- 速率控制不是驱动的一部分
- 每个驱动有自己的速率控制选择算法
- 速率控制填充ieee80211_tx_info速率信息
- 速率控制获取发送状态

从mac80211到驱动

- 驱动方法 (ieee80211_ops)
- mac80211有一些输出函数
- 参考include/net/mac80211.h

7、主要函数

ieee80211_alloc_hw()

- 分配wiphy对象空间 (保证私有数据和硬件私有数据32字节对齐, wiphy包含ieee80211_local和驱动私有数据)
- 初始化wiphy对象 (包括重传次数, RTS门限等等)
- 初始化ieee80211_local (包括重传次数, 工作队列, 接口链表等等)
- 初始化sta_pending_list链表
- 初始化sta_list链表

ieee80211_register_hw()

- 分配int_scan_req数据结构
- 初始化支持接口类型 (包括MONITOR接口)
- 注册wiphy
- 初始化WEP
- 初始化速率控制算法
- 注册STA接口 (默认wlan0)

ieee80211_rx()

- 拷贝skb, 同时在skb头部增加radiotap信息, 传递给所有监听接口
- 如果是数据帧, 根据MAC地址查找station
- 如果station没有找到, 把skb传递给所有接口处理
- 数据帧: 转换成802.3帧格式, 传递给网络协议栈
- 管理帧/控制帧: 传递给MLME

ieee80211_xmit()

- 如果skb来自监听接口, 移除skb头部的radiotap信息

- 进行skb预处理（包括设置QoS优先级，设置分段标志，ACK应答标志等等）
- 选择加密密钥
- 选择速率（ESP8089采用硬件速率控制，所以mac80211速率控制无效）
- 加密（mac80211采用硬件加速，所以mac80211加密无效）
- 通过local->ops->tx()把skb传递给驱动

8、速率控制

Minstrel是mac80211从MadWifi移植过来的速率控制算法，支持多速率重传和提供最好速率。

工作原理

我们定义衡量吞吐量（发包数）的成功，用发送的比特数。

$$\text{Throughput} = \frac{\text{Prob_success_transmission} * \text{Mega bits transmitted}}{\text{time for 1 try of 1 packet to be sent on the air}}$$

这个措施将获取无线接口的最大速率编号来调整传输速度。而且，这表示在优先使用11Mbps速率的情况将不使用1Mbps速率。这个模块将记录所有已发送包的成功结果。通过这个数据，模块就有充分的信息去决定哪个包最成功。但是，需要一个可变参数。去强制模块检查最理想的速率。所以，一些百分比的包使用非正常速率进行发送。

重传序列

一些器件自己已经创建多速率重传序列。比如Atheros 11abg芯片组有四个段。每一段指导硬件采用某些速率来发送当前包，和固定的重传次数。当包发送成功，剩余重传序列被忽略。重传次数的选择是根据期望在26ms内发包出去，或者失败。重传序列是通过两个合理的规则计算的，如果包是一个普通发送包（90%的包）那么重传数是best throughput, next best throughput, best probability, lowest baserate。如果是采样包（10%的包）那么重传数是random lookaround, best throughput, best probability, lowest baserate。表格如下：

Try	Lookaround rate	Normal rate
1	Random lookaround	Best throughput
2	Best throughput	Next best throughput
3	Best probability	Best probability
4	Lowest Baserate	Lowest Baserate

重传数是经过调整的，所以重传序列部分发送时间小于26ms。表格修改如下：

Try	Lookaround rate		Normal rate
	random < best	random > best	
1	Best throughput	Random rate	Best throughput
2	Random rate	Best throughput	Next best throughput
3	Best probability	Best probability	Best probability
4	Lowest Baserate	Lowest Baserate	Lowest Baserate

EWMA

EWMA (Exponential Weighted Moving Average) 是Minstrel速率算法的核心。每秒钟实现10次EWMA计算，每个

速率都会进行计算。计算结果有平滑效果，所以新的结果对于所选择的速率有合理的影响。

作者注：由于水平有限，有些地方不能完全理解原意。敬请谅解并恳请指正。我的联系方式QQ：125548644。

顶

1

踩

0

猜你在找

- iOS8开发技术（Swift版）：iOS基础知识

linux mac80211
- 2016软考软件设计师—基础知识培训视频

linux无线网卡驱动MAC80211架构数据结构
- Struts实战-使用SSH框架技术开发学籍管理系统

mac80211
- 嵌入式Linux高级驱动教程(韦东山2期)

Mac80211定时器
- 全能项目经理训练营

关于mac80211

尚品宅配

全屋家具定制

衣柜“隐身”

房间果然大了

定制家具，给家一个礼物。CUSTOM FURNITURE, A GIFT TO THE FAMILY.

去看看

GO>




查看评论

4楼 [eveduo](#) 2016-04-07 15:33发表




请问 图2-1 系统框架 中 nl80211 和 cfg80211_ops 是不是位置反了？

3楼 [Y蓝羽Y](#) 2016-03-23 17:33发表




感谢博主总结，学习了

2楼 [yutengao1987](#) 2015-11-06 13:44发表



学习了 帧怎么抓呢

1楼 [lvy-cx](#) 2014-10-26 19:18发表



学习了 博主，请问下在哪里能捕捉到控制帧和管理帧？

Re: [13459104362](#) 2015-01-12 10:45发表



回复lvy-cx：开启monitor模式就可以抓到。

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

- 全部主题
- Hadoop
- AWS
- 移动游戏
- Java
- Android
- iOS
- Swift
- 智能硬件
- Docker
- OpenStack
- VPN
- Spark
- ERP
- IE10
- Eclipse
- CRM
- JavaScript
- 数据库
- Ubuntu
- NFC
- WAP
- jQuery
- BI
- HTML5
- Spring
- Apache
- .NET
- API
- HTML
- SDK
- IIS
- Fedora
- XML
- LBS
- Unity
- Splashtop
- UML
- components
- Windows Mobile
- Rails
- QEMU
- KDE
- Cassandra
- CloudStack
- FTC
- coremail
- OPhone
- CouchBase
- 云计算
- iOS6
- Rackspace
- Web App
- SpringSide
- Maemo

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [银行汇款帐号](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

[网站客服](#) [杂志客服](#) [微博客服](#) [webmaster@csdn.net](#) 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持

京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved 