百度一下

您查询的关键词是： **iw 使用** 以下是该网页在北京时间 2017年02月04日 21:47:37 的快照；

如果打开速度慢，可以尝试快速版；如果想更新或删除快照，可以投诉快照。

百度和网页 http://blog.csdn.net/a_tu_/article/details/51982976 的作者无关，不对其内容负责。百度快照谨为网络故障时之索引，不代表被搜索网站的即时页面。

# a_tu_的专栏

- 目录视图
- 摘要视图
- 订阅

# iw 命令

2016-07-21 15:05 368人阅读 评论(0) 收藏 举报

分类：

openwrt学习（13）

作者同类文章X

## 关于 iw

**iw** 是一种新的基于 nl80211 的用于无线设备的CLI配置实用程序。它支持最近已添加到内核所有新的驱动程序。采用无线扩展接口的旧工具iwconfig已被废弃，强烈建议切换到 **iw** 和 nl80211。

像Linux内核的其余部分，**iw** 仍处于开发阶段。功能被随时添加。 **iw** 的唯一文档就是此页和“**iw** help”的输出。 请帮助扩大这个页面。

有一个页面列出iwconfig和**iw**的用例：替换 iwconfig。

## 获得 iw

**iw**发布压缩包可以从这里获取： http://kernel.org/pub/software/network/**iw**/.

另外，你也可以从 git 下载 **iw**: http://git.kernel.org/?p=linux/kernel/git/jberg/**iw**.git.

## 编译要求

- libnl >= libnl1
- libnl-dev >= libnl-dev-1
- pkg-config

为了**使用** **iw** 你需要有libnl，第一个工作版本是1.0 pre8，因为此版本中引入了 genl， Generic Netlink，它是nl80211所依赖的。如果你的发行版的libnl是一个错误的版本，那么你就必须下载并自己编译libnl（http://www.infradead.org/~tgr/libnl/）。

## 帮助

只要在命令行输入

```
iw help
```

它会打印出它所支持的命令。

### 获取设备的功能

使用以下方法来获得所有设备的功能，如带宽信息（2.4GHz，和5GHz），和802.11n的信息：

```
iw list
```

### 扫描

```
iw dev wlan0 scan
```

### 监听事件

只要使用

```
iw event
```

调试时观察auth/assoc/deauth/disassoc帧可能有帮助，使用

```
iw event -f
```

有时时间信息也是有用的：

```
iw event -t
```

### 获得链路状态

为了确定您是否连接到一个AP上，如果你是最后一个使用的发送速率，您可以使用下面的命令。

关联到传统（非802.11n标准）的AP的输出示例：

```
iw dev wlan0 link
Connected to 04:21:b0:e8:c8:8b (on wlan0)
        SSID: attwifi
        freq: 2437
        RX: 2272 bytes (18 packets)
        TX: 232 bytes (3 packets)
        signal: -57 dBm
        tx bitrate: 36.0 MBit/s
```

关联到802.11n的AP的输出示例：

```
iw dev wlan0 link
Connected to 68:7f:74:3b:b0:01 (on wlan0)
        SSID: tesla-5g-bcm
        freq: 5745
        RX: 30206 bytes (201 packets)
        TX: 4084 bytes (23 packets)
        signal: -31 dBm
        tx bitrate: 300.0 MBit/s MCS 15 40Mhz short GI
```

当输出没有连接到一个AP的例子：

```
iw dev wlan0 link
Not connected.
```

这会发生在当你没有连接到一个AP的情况下。 要连接到一个AP可以**使用** **iw** connect ， 如果连接要求：

- 无加密
- **使用**WEP加密

如果你需要连接到一个AP要求**使用**WPA或WPA2，那么你必须**使用**wpa_supplicant 。

## 建立基本连接

你可以**使用****iw**直接连接到一个AP上，当且仅当AP：

- 无加密
- **使用**WEP加密

然而应该指出的是，如果你断开AP的连接时，这可以在一个繁忙的环境中频繁发生，你将需要重新发出命令。 如果你不想这样做，你可以**使用**wpa_supplicant，它会在你断开连接后自动尝试重新连接。

如果您选择自己处理断开连接，你可以**使用** **iw** connect 如下所示。

如要连接到已禁用加密的AP，这里它的SSID是foo ：

```
iw wlan0 connect foo
```

假设你有两个AP SSID 都是 foo ，你知道你要连接的是在 2432 频道，你可以指定**使用**的频率：

```
iw wlan0 connect foo 2432
```

要连接到**使用**WEP的AP，则可以**使用**：

```
iw wlan0 connect foo keys 0:abcde d:1:0011223344
```

## 获取station统计数据

要获取station 的统计信息，如发送/接收的字节，最后发送的比特率（包括MCS率），你可以执行下面的命令：

```
$ iw dev wlan1 station dump
Station 12:34:56:78:9a:bc (on wlan0)
        inactive time:  304 ms
        rx bytes:       18816
        rx packets:     75
        tx bytes:       5386
        tx packets:     21
        signal:         -29 dBm
        tx bitrate:     54.0 MBit/s
```

## 获得station对应的peer统计信息

如果你想获得 station 对应的 peer的统计信息，你可以**使用**下面的命令：

```
sudo iw dev wlan1 station get <peer-MAC-address>
```

在STA的情况下，上述的<peer-MAC-address>将是你的AP的MAC地址。

## 修改传输比特率

**iw** 支持修改发送的比特率，假托传统和 HT MCS速率。 这是通过设定允许的比特率掩码来实现，你也可以清除该掩码。

### 修改TX传统的比特率

您可以设定用于**使用**某些传统的比特率传输的性能。 例如：

```
iw wlan0 set bitrates legacy-2.4 12 18 24
```

以下是如何使能一些人所说的“Purge G”来禁用802.11b 关联：

```
iw wlan0 set bitrates legacy-2.4 6 12 24
```

### 修改tx HT MCS的比特率

设置**使用**MCS率传输的能力是通过让你指定频段和MCS率来实现。 注意，是否该设备实际上监听你的请求将根据设备驱动程序和固件的配合而变化。 例如：

```
iw dev wlan0 set bitrates mcs-5 4
```

```
iw dev wlan0 set bitrates mcs-2.4 10
```

清除所有 tx 比特率和设置的东西来恢复正常：

```
iw dev wlan0 set bitrates mcs-2.4
iw dev wlan0 set bitrates mcs-5
```

### 设置传输功率

可以通过**使用**各自的phy的任一设备接口名称来设置 txpower 。

```
iw dev <devname> set txpower <auto|fixed|limit> [<tx power in mBm>]
iw phy <phyname> set txpower <auto|fixed|limit> [<tx power in mBm>]
```

### 省电

为在默认情况下使能 power save，你可以**使用**：

```
sudo iw dev wlan0 set power_save on
```

对于mac80211驱动程序，这意味着动态节电模式被启用。

查询当前的节电设定，您可以**使用**：

```
iw dev wlan0 get power_save
```

### **使用iw**添加接口

有几种模式可以支持。 支持的模式包括：

- monitor
- managed [also station]
- wds
- mesh [also mp]
- ibss [also adhoc]

要查看这些说明，请阅读我们的模式文档 。

例如，要添加一个 monitor 接口：

```
iw phy phy0 interface add moni0 type monitor
```

where you can replace `monitor` by anything else and `moni0` by the interface name, and need to replace`phy0` by the PHY name for your hardware (usually phy0 will be correct unless you hotplugged or reloaded any modules.) If your udev is configured incorrectly, the newly created virtual interface may be renamed by it right away, use `ip link` to list all interfaces.

Note that in case you want to monitor 802.11n you will need to specify channel width (20 or 20/40MHz) and in case of 20/40MHz if the upper or lower channel is being used. To do so you would use:

```
iw dev <devname> set freq <freq> [HT20|HT40+|HT40-]
```

or

```
iw phy <phyname> set freq <freq> [HT20|HT40+|HT40-]
```

You can also specify channel instead of frequency:

```
iw phy <phyname> set channel <channel> [HT20|HT40+|HT40-]
iw dev <devname> set channel <channel> [HT20|HT40+|HT40-]
```

To create a new managed mode interface you would use:

```
iw phy phy0 interface add wlan10 type managed
```

Note that the interface is automatically put into AP mode when using hostapd.

### Modifying monitor interface flags

You can customize the type of monitor interface you create. This can be very useful for debugging purposes on end user systems. For example suppose you want to help a user you can take advantage of the fact that a monitor interface in mac80211 uses radiotap to pass up to userspace additional data. Say we want to help a user fish out data without affecting the device's performance by setting it it to a full monitor interface an monitor interface with no additional monitor flags can be created as follows:

```
iw dev wlan0 interface add fish0 type monitor flags none
```

You can then request the user to use tcpdump on a session:

```
tcpdump -i fish0 -s 65000 -p -U -w  /tmp/fishing.dump
```

The nice thing about these type of alternative monitor interfaces is you can further extend radiotap even with vendor extensions to add more data to radiotap to help debug device specific features.

Keep in mind this requires drivers to honor mac80211's flag requests strictly, so drivers like ath5k and ath9k which still enable flags based on operation mode need to be fixed to take advantage of this.

#### Monitor flags possible

The following are flags you can specify:

- none
- fcsfail
- plcpfail
- control
- otherbss
- cook

## Deleting interfaces with **iw**

The command line is:

```
iw dev moni0 del
```

Where "moni0" was the virtual interface interface that was created with the first command

### Virtual vif support

There is a dedicated section for virtual vif support, see the **iw** vif page.

### Setting frequency with **iw**

The command line is:

```
iw dev wlan0 set freq 2412 [HT20|HT40+|HT40-]
```

### Setting channel with **iw**

The command line is:

```
iw dev wlan0 set channel 1 [HT20|HT40+|HT40-]
```

### Updating your regulatory domain

The command line is:

```
iw reg set alpha2
```

Where "alpha2" is the ISO/IEC 3166 alpha2 country code. The information used and set comes from our regulatory infrastructure.

You can also use the latest wpa_supplicant (as of 0.6.7) now to change your regulatory domain, to do so just add a "COUNTRY=US" entry into your configuration for example.

### Creating and inspecting Mesh Point interfaces with **iw**

You may add a mesh interface to drivers that support Mesh Point operation. Mesh Point interfaces have a mesh_id parameter which may be up to 32 bytes long. For example, to add an interface "mesh0" to device phy0 with mesh_id "mymesh",

```
iw phy phy0 interface add mesh0 type mp mesh_id mymesh
```

Mesh Point interfaces, by default, are configured on Channel 1. Mesh Point operation begins when the interface is brought up. In the default configuration, Mesh Point interfaces will automatically detect and attempt to create Peer Links with other Mesh Points (peers) having the same mesh ID. Use the station list and station statistics to see the peer list and Peer Link status.

After sending traffic (ex: pinging another mesh node), you may wish to see a list of Mesh Paths:

```
iw dev mesh0 mpath dump
```

Please see the open80211s.org HOWTO for further details on Mesh Point related commands and their output, as well as more examples. **iw** also provides commands for advanced Mesh Point configuration. These are documented in the Advanced Tinkering section of the open80211s HOWTO.

### Setting up a WDS peer

WDS mode is a non-standard extension to the IEEE 802.11 standard to allow transparent Ethernet bridging on the station and to implement seamingless hand-over for wireless clients roaming between different access points. Due to its non-standard nature, WDS is often implemented differently in wireless drivers and vendor firmwares making them incompatible with each other. In order to use WDS, one should use the same hardware and software on all deployed wireless devices to maintain compatibility.

To create a WDS peer you will first need to create an interface of WDS type, and then set the peer:

```
iw phy phy0 interface add wds0 type wds
iw dev wds0 set peer <MAC address>
```

In order for this to work the driver must implement the cfg80211 callback set_wds_peer(). mac80211 implements this callback, so the respective mac80211 driver would just need to support WDS type interfaces. What WDS will do is replace the first address on the 802.11 header with the peer address when TXing frames. Instead of using WDS though you may want to consider using 4-address mode described below if you have control over the software running on the AP and respective clients/peers connected.

### Using 4-address for AP and client mode

In some situations it might be useful to run a network with an Access Point and multiple clients, but with each client bridged to a network behind it. For this to work, both the client and the AP need to transmit 4-address frames, containing both source and destination MAC addresses. 4-address mode is howOpenWrt supports WDS mode for mac80211 drivers, that is if you enable *wds* option on your OpenWrt OpenWrt wireless configuration you will end up using 4-address mode. 4-address mode is not compatible with other WDS implementations, ie, you'll need all endpoints using this mode in order for WDS to work appropriately.

Linux wireless has support for 4-address mode for AP and STAs but each driver needs to define this capability explicitly. All mac80211 drivers support 4-address mode if AP or STA modes of operation are supported respectively.

On the AP side you can enable 4-address frames for individual clients by isolating them in separate AP VLANs which are configured in 4-address mode. Such an AP VLAN will be limited to one client only, and this client will be used as the destination for all traffic on its interface, regardless of the destination MAC address in the packet headers. The advantage of this mode compared to regular WDS mode is that it's easier to configure and does not require a static list of peer MAC addresses on any side. 4-address mode is incompatible with WDS.

To enable 4-address mode when creating an interface you should add *4addr on*, for example:

```
iw phy phy0 interface add moni0 type managed 4addr on
```

When the client side interface is included in a bridge, add *-b <bridge_interface>* when runningwpa_supplicant.

In hostapd you can enable this with the flag on hostapd.conf:

```
wds_sta=1
```

Please note 4-address mode is currently broken on 3.9 because of commit 576eb62598f10c8c7fd75703fe89010cdcfff596 , this topic is currently being addressed on the mailing lists for a resolution.

### Creating packet coalesce rules

In most cases, host that receives IPv4 and IPv6 multicast/broadcast packets does not do anything with these packets. Therefore the reception of these unwanted packets causes unnecessary processing and power consumption.

Packet coalesce feature helps to reduce number of receive interrupts to host by buffering these packets in firmware/hardware for some predefined time. Receive interrupt will be generated when one of the following events occur.

- Expiration of hardware timer whose expiration time is set to maximum coalescing delay of matching coalesce rule.
- Coalescing buffer in hardware reaches it's limit.
- Packet doesn't match any of the configured coalesce rules.

To view coalesce configuration support information, you can use '**iw** phy0 info'. Here is an example output:

```
Coalesce support:
         * Maximum 8 coalesce rules supported
         * Each rule contains upto 4 patterns of 1-4 bytes,
           maximum packet offset 50 bytes
         * Maximum supported coalescing delay 100 msecs
```

You need to configure following parameters for creating a coalesce rule.

- Maximum coalescing delay
- List of packet patterns which needs to be matched
- Condition for coalescence. pattern 'match' or 'no match'

Multiple such rules can be provided through a configuration file.

To enable coalesce feature using rules listed in coalesce.conf file, you can use:

```
iw phy phy0 enable coalesce.conf
```

Where coalesce.conf contains:

```
delay=25
condition=0
patterns=8+34:xx:ad:22,10+23:45:67,59:33:xx:25,ff:ff:ff:ff
delay=40
condition=1
patterns=12+00:xx:12,23:45:67,46:61:xx:50
```

To display current coalesce configuration, you can use:

```
$ iw phy phy0 coalesce show
Coalesce is enabled:
Rule - max coalescing delay: 25msec condition:match
 * packet offset: 8 pattern: 34:--:ad:22
 * packet offset: 10 pattern: 23:45:67
 * packet offset: 0 pattern: 59:33:--:25
 * packet offset: 0 pattern: ff:ff:ff:ff
Rule - max coalescing delay: 40msec condition:not match
 * packet offset: 12 pattern: 00:--:12
 * packet offset: 0 pattern: 23:45:67
 * packet offset: 0 pattern: 46:61:--:50
```

To disable coalesce feature, you can use:

```
iw phy phy0 coalesce disable
```

'**iw** display' output when coalesce is not configured:

```
$ iw phy phy0 coalesce show
Coalesce is disabled.
```

顶
1

踩
0

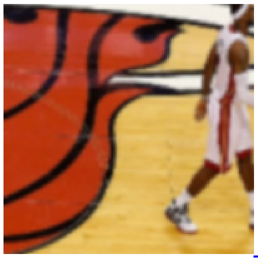
- 上一篇iptables规则的查看、添加、删除和修改
- 下一篇openwrt 内核 打补丁

## 参考知识库

猜你在找

查看评论
* 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

个人资料



a_tu_

- 访问：46906次
- 积分：1019

- 等级： BLOG 4
- 排名：千里之外
- 原创：46篇
- 转载：48篇
- 译文：0篇
- 评论：11条

文章搜索

搜索

文章分类

- 嵌入式(8)
- linux(45)
- curl(7)
- socket编程(5)
- json(0)
- usb wifi模块移植(2)
- linux-usb学习(7)
- sh工具(1)
- 华为3G（MU709） 模块(3)
- openwrt学习(14)
- 路由(2)
- git相关(2)

文章存档

- 2016年12月(1)
- 2016年10月(2)
- 2016年09月(1)
- 2016年08月(1)
- 2016年07月(8)
- 2016年06月(4)
- 2016年05月(2)
- 2016年04月(5)
- 2016年01月(1)
- 2015年10月(3)
- 2015年08月(3)
- 2015年06月(3)
- 2015年05月(1)
- 2015年04月(1)
- 2015年03月(6)
- 2015年02月(4)
- 2015年01月(7)
- 2014年12月(14)
- 2014年11月(9)
- 2014年10月(13)
- 2014年09月(5)

阅读排行

- openwrt 添加4G拨号（qmi驱动）(3414)
- 关于加载驱动rtl8188EUS（8188eu.ko）到内核3.15出现段错误(2197)
- Linux USB虚拟串口设备(2061)
- LINUX 自定义USB Gadget HID 设备(1742)
- 华为3G模块拨号上网(1713)
- AT指令建立TCP/IP 链接，并自收发数据(1599)
- linux 开发板（Linux USB Gadget） 模拟u盘(1491)
- 用curl下载一张图片并保存(1257)
- AT指令发送短信流程及接听、拨打电话(935)
- rtl8723wifi（蓝牙模块）模块移植(878)

评论排行

- openwrt 添加4G拨号（qmi驱动）(5)

- wpa_supplicant2.2 移植(2)
- 关于加载驱动rtl8188EUS（8188eu.ko）到内核3.15出现段错误(2)
- Sublime Text3 中文输入（不需要安装gtk及其他库）(1)
- 华为3G模块拨号上网(1)
- ./configure 的参数解析(0)
- "undefined reference to" 问题解决方法(0)
- Notepad++ 16进制编辑功能(0)
- git命令之git tag 给当前分支打标签(0)
- curl_easy_setopt函数介绍(0)

推荐文章

- * 而立之年——三线城市程序员的年终告白
- * Java集合框架中隐藏的设计套路
- * Python脚本下载今日头条视频(附加Android版本辅助下载器)
- * 人工智能的冷思考
- * React Native 实战系列教程之热更新原理分析与实现

最新评论

- openwrt 添加4G拨号（qmi驱动）
  sunqinglin4826: 楼主您好，可以麻烦问您下，您make menuconfig的时候，都选择了哪些软件包吗~谢谢谢谢
- openwrt 添加4G拨号（qmi驱动）
  a_tu_: 你试试echo 1c9e 9b01 > /sys/bus/usb/drivers/qmi_wwan/...
- openwrt 添加4G拨号（qmi驱动）
  qq_22340085: 楼主你好， 我这编译出了 cdc-wdm模块， 插上模块后只看到ttyUSB0-ttyUSB4， 但...
- openwrt 添加4G拨号（qmi驱动）
  a_tu_: @baidu_35299190:有是有，得改一下
- openwrt 添加4G拨号（qmi驱动）
  baidu_35299190: 你好 你这个驱动是怎么配置的？openwrt里面有这个4g模块的驱动吗？
- 华为3G模块拨号上网
  qq_28914249: 个人觉得这个行业和产品，在未来是有很大的发展潜力的，很看好这方面，如果可以像那个隐形云盘一样搞个加密...
- wpa_supplicant2.2 移植
  a_tu_: 当然可以用啊
- wpa_supplicant2.2 移植
  qxq125478: 可以用吗？好好整理下，有点乱
- Sublime Text3 中文输入（不需要安装gtk及其他库）
  qxq125478: 下载还要分吗？，不会是为自己的资源打广告的吧Σ( ° △ ° |||)⌇
- 关于加载驱动rtl8188EUS（8188eu.ko）到内核3.15出现段错误
  a_tu_: @wt86024:这两个工具是交叉编译出来的，iwconfig是从wireless_tools_29...