How to use packet injection with mac80211

mac80211 now allows arbitrary packets to be injected down any Monitor Mode interface from userland. The packet you inject needs to be composed in the following format:

```
[ radiotap header  ]
[ ieee80211 header ]
[ payload ]
```

The radiotap format is discussed in ./Documentation/networking/radiotap-headers.rst.

Despite many radiotap parameters being currently defined, most only make sense to appear on received packets. The following information is parsed from the radiotap headers and used to control injection:

- IEEE80211_RADIOTAP_FLAGS

IEEE80211_RADIOTAP_F_FCS	FCS will be removed and recalculated
IEEE80211_RADIOTAP_F_WEP	frame will be encrypted if key available
IEEE80211_RADIOTAP_F_FRAG	frame will be fragmented if longer than the current fragmentation threshold.

- IEEE80211_RADIOTAP_TX_FLAGS

IEEE80211_RADIOTAP_F_TX_NOACK	frame should be sent without waiting for an ACK even if it is a unicast frame

- IEEE80211_RADIOTAP_RATE

  legacy rate for the transmission (only for devices without own rate control)

- IEEE80211_RADIOTAP_MCS

  HT rate for the transmission (only for devices without own rate control). Also some flags are parsed

IEEE80211_RADIOTAP_MCS_SGI	use short guard interval
IEEE80211_RADIOTAP_MCS_BW_40	send in HT40 mode

- IEEE80211_RADIOTAP_DATA_RETRIES

  number of retries when either IEEE80211_RADIOTAP_RATE or IEEE80211_RADIOTAP_MCS was used

- IEEE80211_RADIOTAP_VHT

  VHT mcs and number of streams used in the transmission (only for devices without own rate control). Also other fields are parsed

  flags field
  IEEE80211_RADIOTAP_VHT_FLAG_SGI: use short guard interval
  bandwidth field
  - 1: send using 40MHz channel width
  - 4: send using 80MHz channel width
  - 11: send using 160MHz channel width

The injection code can also skip all other currently defined radiotap fields facilitating replay of captured radiotap headers directly.

Here is an example valid radiotap header defining some parameters:

```
0x00, 0x00, // <-- radiotap version
0x0b, 0x00, // <- radiotap header length
0x04, 0x0c, 0x00, 0x00, // <-- bitmap
0x6c, // <-- rate
0x0c, //<-- tx power
0x01 //<-- antenna
```

The ieee80211 header follows immediately afterwards, looking for example like this:

```
0x08, 0x01, 0x00, 0x00,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x13, 0x22, 0x33, 0x44, 0x55, 0x66,
0x13, 0x22, 0x33, 0x44, 0x55, 0x66,
0x10, 0x86
```

Then lastly there is the payload.

After composing the packet contents, it is sent by send()-ing it to a logical mac80211 interface that is in Monitor mode. Libpcap can also be used, (which is easier than doing the work to bind the socket to the right interface), along the following lines::

```
ppcap = pcap_open_live(szInterfaceName, 800, 1, 20, szErrbuf);
```

```
...
r = pcap_inject(ppcap, u8aSendBuffer, nLength);
```

You can also find a link to a complete inject application here:

https://wireless.wiki.kernel.org/en/users/Documentation/packetspammer

Andy Green <andy@warmcat.com>