Linux IEEE 802.15.4 implementation

Introduction

The Linux-ZigBee project goal is to provide complete implementation
of IEEE 802.15.4 / ZigBee / 6LoWPAN protocols. IEEE 802.15.4 is a stack
of protocols for organizing Low-Rate Wireless Personal Area Networks.

Currently only IEEE 802.15.4 layer is implemented. We have choosen
to use plain Berkeley socket API, the generic Linux networking stack
to transfer IEEE 802.15.4 messages and a special protocol over genetlink
for configuration/management

Socket API

```
int sd = socket(PF_IEEE802154, SOCK_DGRAM, 0);
.....
```

The address family, socket addresses etc. are defined in the
include/net/af_ieee802154.h header or in the special header
in our userspace package (see either linux-zigbee sourceforge download page
or git tree at git://linux-zigbee.git.sourceforge.net/gitroot/linux-zigbee).

One can use SOCK_RAW for passing raw data towards device xmit function. YMMV.

MLME - MAC Level Management

Most of IEEE 802.15.4 MLME interfaces are directly mapped on netlink commands.
See the include/net/nl802154.h header. Our userspace tools package
(see above) provides CLI configuration utility for radio interfaces and simple
coordinator for IEEE 802.15.4 networks as an example users of MLME protocol.

Kernel side

Like with WiFi, there are several types of devices implementing IEEE 802.15.4.
1) 'HardMAC'. The MAC layer is implemented in the device itself, the device
   exports MLME and data API.
2) 'SoftMAC' or just radio. These types of devices are just radio transceivers
   possibly with some kinds of acceleration like automatic CRC computation and
   comparation, automagic ACK handling, address matching, etc.

Those types of devices require different approach to be hooked into Linux
kernel.

HardMAC

See the header include/net/ieee802154_netdev.h. You have to implement Linux net_device, with .type = ARPHRD_IEEE802154. Data is exchanged with socket family code via plain sk_buffs. On skb reception skb->cb must contain additional info as described in the struct ieee802154_mac_cb. During packet transmission the skb->cb is used to provide additional data to device's header_ops->create function. Be aware, that this data can be overriden later (when socket code submits skb to qdisc), so if you need something from that cb later, you should store info in the skb->data on your own.

To hook the MLME interface you have to populate the ml_priv field of your net_device with a pointer to struct ieee802154_mlme_ops instance. All fields are required.

We provide an example of simple HardMAC driver at drivers/ieee802154/fakehard.c

SoftMAC
=======

We are going to provide intermediate layer implementing IEEE 802.15.4 MAC in software. This is currently WIP.

See header include/net/mac802154.h and several drivers in drivers/ieee802154/.