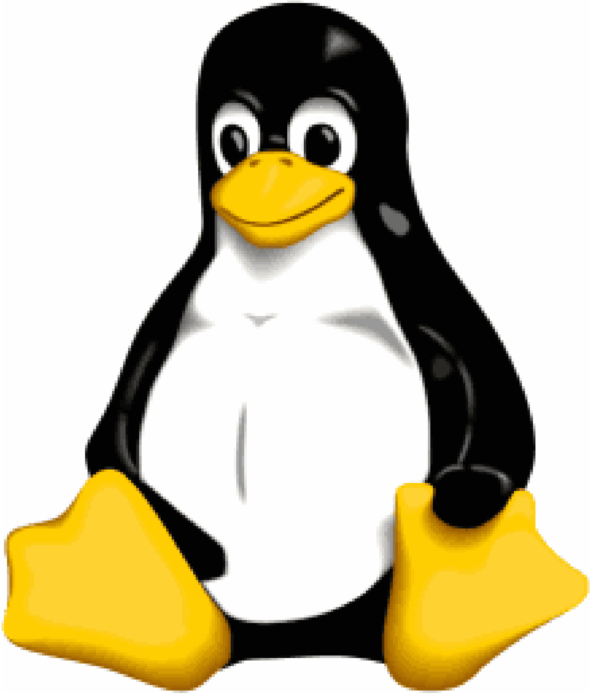
SocketCAN
The official CAN API of the Linux Kernel

Automotive Linux Summit Fall 2013

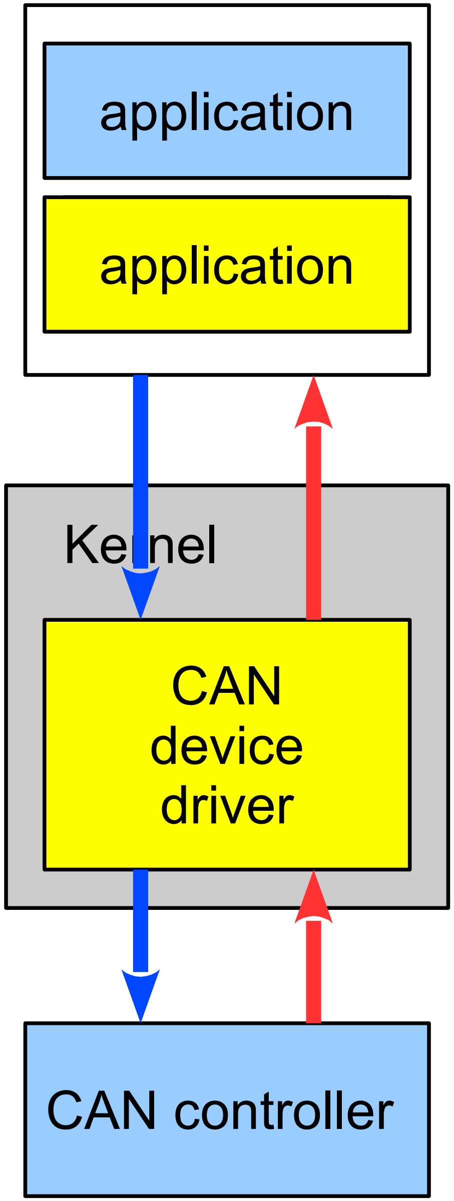
Marc Kleine-Budde
Pengutronix
<mkl@pengutronix.de>

Overview

- History of CAN in Linux
- Linux networking subsystem
- CAN device driver
- Userspace example

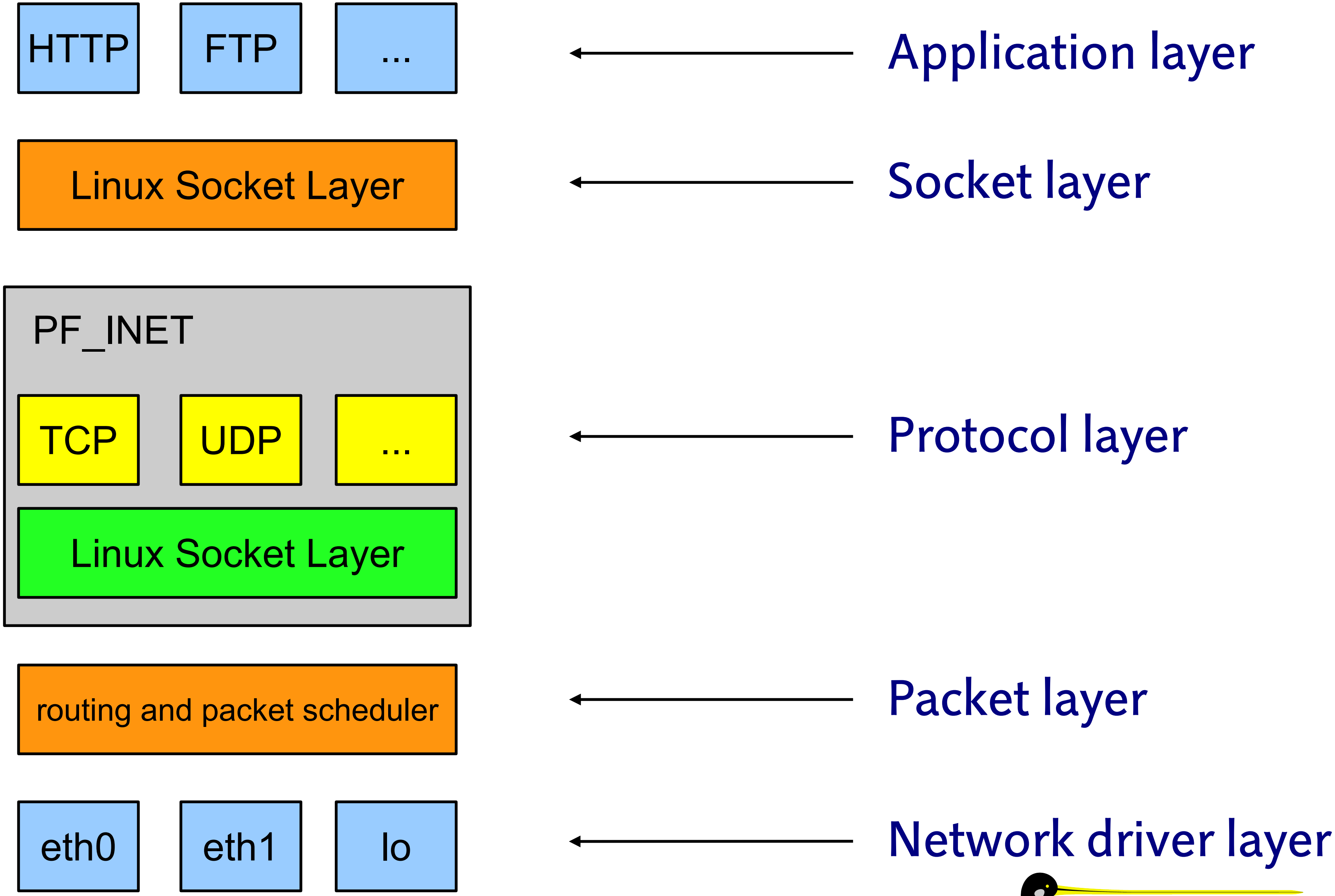


CAN access in Linux – before SocketCAN

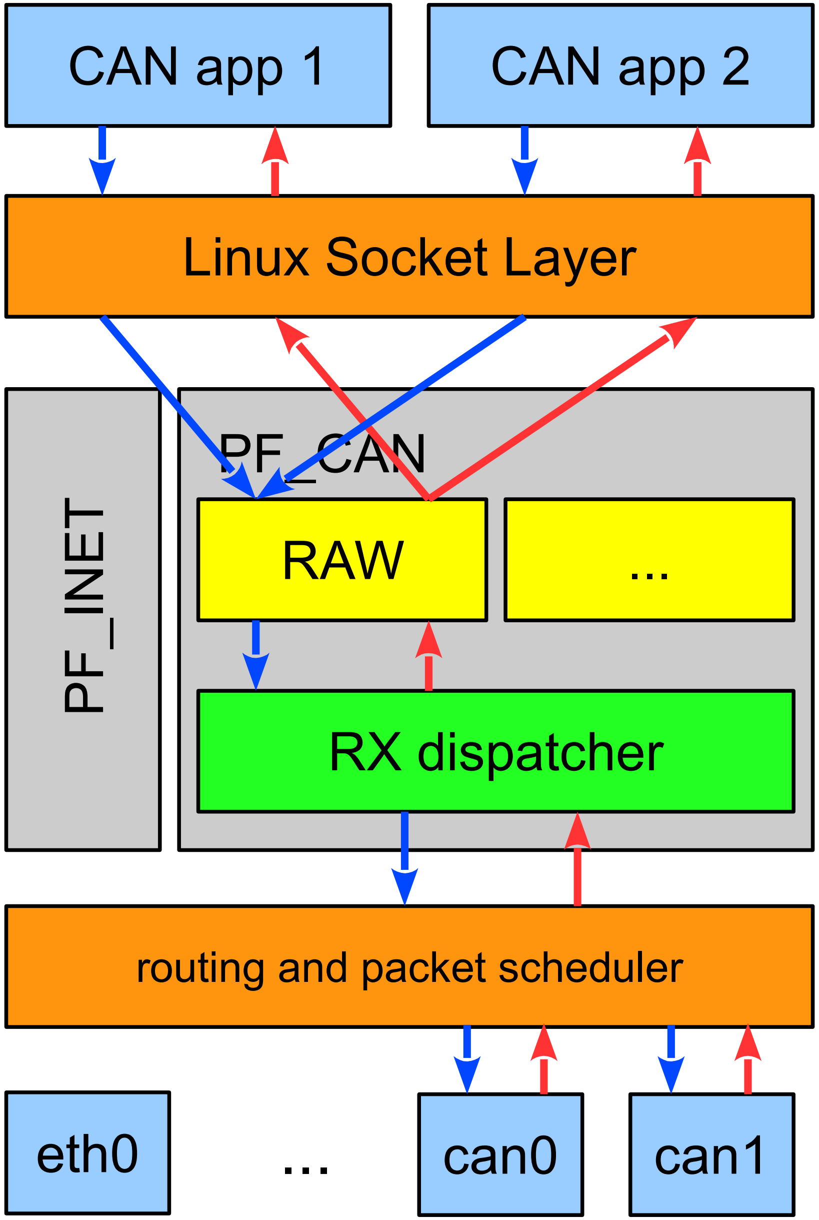


- No standard Linux CAN driver model, all character device based
- Only single application at a time
- Higher level protocols and filtering have to be implemented in application
- CAN hardware vendor provide own driver
- Change of hardware vendor urges adaptation of CAN application

The Linux networking subsystem



A socket-based approach



CAN applications

SocketCAN Protocol layer, new protocol family: **PF_CAN**

Different CAN protocol plugins: **CAN_RAW**

RX dispatcher: distributes CAN frames to all registered protocols

CAN drivers

Multi-application support

- For **CAN_RAW** each socket can specify a filter list
- RX dispatcher implements complex filtering
- Received CAN frames are transmitted to all CAN protocols that have a matching filter
- Local originated CAN frames are looped back into the RX queue

Multi-application support – Why to loopback?

- Consider two embedded systems, each running a SocketCAN application
- They exchange messages via the CAN bus
- If these applications run on the same system, they still have to see each other's CAN messages
- Put CAN frame into RX queue after transmission has been completed
- For best results, to preserve sequence of frames, do echoing in TX complete interrupt handler

A socket-based approach – the advantages

- Use of existing and established POSIX socket API
- New protocol family **PF_CAN** is developed against established abstractions
- Communication protocols and filtering can be implemented inside the kernel
- CAN network device driver implements standard Ethernet driver model
- Support for multi-user and multi-application possible

Drawbacks and limitations

- More memory overhead, the Linux network stack has been designed for much larger Ethernet frames:
  64 Bytes (min. Ethernet frame) vs.
  8 byte (max. CAN frame)
- The “packet scheduler” is a shared resource among all networking devices
- Heavy Ethernet traffic can lead to delays in CAN traffic.
- No Support for hardware filtering (yet)

CAN networking device drivers

- Initialize and configure hardware
- Receive incoming CAN frames from hardware and push them into upper layer
- Obtain outgoing frame from upper layer and transmit to wire
- Almost identical to Ethernet drivers, but handle CAN instead of Ethernet frames. Make use of existing Ethernet driver model!
- Define bit rate constraints of CAN hardware in clock rate independent way

Applications and the CAN_RAW protocol

- Simplest method to access the CAN bus
- Programming interface similar to character device drivers, transfer whole CAN frames
- First create a **CAN_RAW** socket, then **bind()** to a CAN interface. Use standard systems calls to read and write CAN frames

SocketCAN – struct can_frame

```
/* special address description flags for the CAN_ID */

#define CAN_EFF_FLAG  0x80000000U/* EFF/SFF is set in the MSB */
#define CAN_RTR_FLAG  0x40000000U/* remote transmission request */
#define CAN_ERR_FLAG  0x20000000U/* error frame */

struct can_frame {
    canid_tcan_id;    /* 32 bit CAN_ID + EFF/RTR/ERR flags */
    __u8   can_dlc;   /* data length code: 0 .. 8 */
    __u8   data[8]    __attribute__((aligned(8)));
};
```

Conclusion + Outlook

- Multi-application + Multi-user POSIX socket API to send and receive raw CAN frames

- Standard driver model known from Ethernet drivers

- Kernel internal infrastructure to filter, send and receive CAN frames to implement more complex protocols

- More CAN protocols
  - BCM – Broadcast Manager
  - CANGW – CAN Gateway
  - ISOTP – ISO 15765-2
  - J1939

Questions?

Thanks!

More information:

- Inside the Kernel: Documentation/networking/can.txt
- Mailing-list: linux-can@vger.kernel.org
- Project Homepage: http://gitorious.org/linux-can
  - Upstream git
  - Userspace tools
  - Support for older Kernels