



## Tracing the Way of Data in a TCP Connection through the Linux Kernel

## Summary:

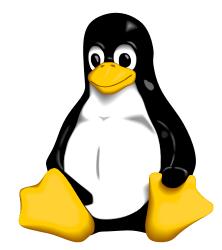
The following figure gives an overview of the linux kernel TCP/IP internals and the involved functions. For clear arrangement all function calls for locking and unlocking of data structures have been left out. The function descriptions are summarized. Often these functions call several subfunctions to do their work. Also detailed call stack semantics are omitted. So when sock\_sendmsg() is placed below sys\_send() this means it gets called after sys\_send() not necessarily by sys\_send().

**User Space** 

send()

**Kernel Space** 

sys\_send()



sock\_sendmsg()

tcp\_sendmsg()

ip\_queue\_xmit()

nf\_hook\_slow()

**Kernel Space** 

e1000\_xmit\_frame()

**Network Device** 

send() – is one of the many ways to put data
into a network socket from userspace. Others
 are write(), sendmsg() or fwrite()

tcp\_sendmsg() – Builds the TCP Header, checksum and handles tcp protocol details.

nf\_hook\_slow () – Here nf stands for netfilter.

Netfilter is the "firewall" subystem of the kernel. Here the ip packet to send is checked against various filter rules.

e1000\_xmit\_frame() – is function of the linux device driver for the intel e1000 network adapter used in the experiment. It takes pointers to the headers and data and builds the final ethernet frame.

## Measurement Method (ftrace):

For these Measurements the Linux Function Tracing Framework ftrace has been used. A recommendable and easy interface to the ftrace kernel facility is the trace-cmd command-line utility. For tracing all kernel calls happening on behalf of process <x> use:

"trace-cmd record -p function\_graph -P <x>"

Richard Sailer

richard.willi.sailer@student.uni-augsburg.de

Full trace log: https://github.com/richi235/ftrace-seminar-paper

