



Breaking Open Linux Switching Drivers

Andy Gospodarek

Cumulus Networks

16 Feb 2015

netdev01

Agenda

- Why
- History
- Design proposal
- Future possibilities

Why am I here?

**Linux kernel should enable others to
create the next generation of forwarding
devices**

**Integrate support for offload hardware
directly into the the Linux kernel**

Hardware Platform History

**Market dominated by switch and
router vendors providing expensive
proprietary solutions**

**Proprietary software running on switches
and routers was not open for developers
and users to enhance**

**Today's hardware platforms are
significantly higher-performance and more
generally available**

**Spare CPU cycles are available for
applications to run directly on the switch**

**Bare-metal platforms are now appealing
and available to commercial Linux
vendors, developers, and users**

Software History

**15+ years with Linux as a viable OS for host
processor on switches/routers**

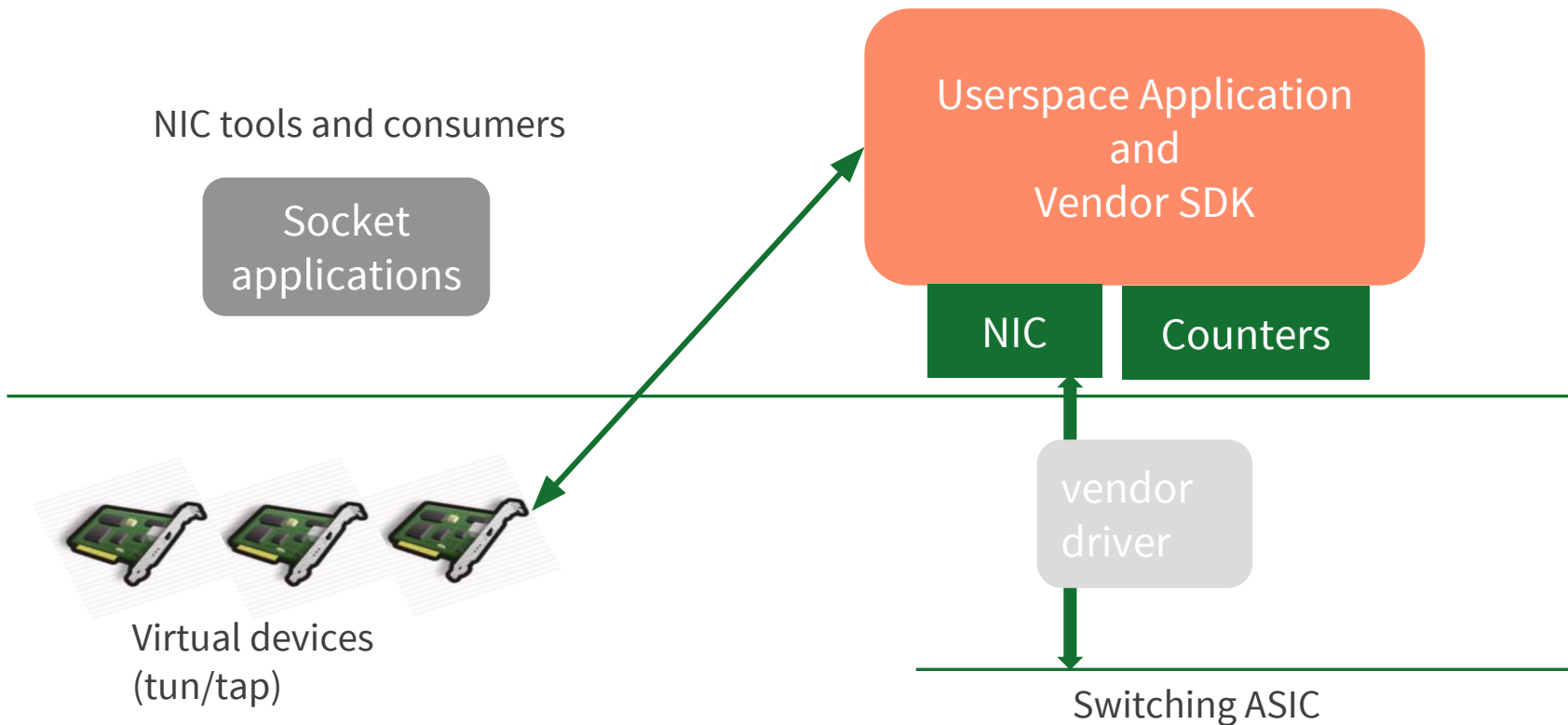
10+ years Linux “support” by ASIC vendors

Basic in-kernel switching/offload layer support in v3.19

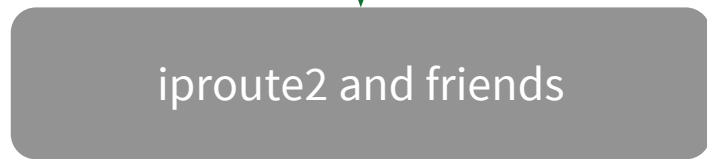
**Software architecture to control ASICs
has not fundamentally
changed in the last decade**

What exactly does that look like?

Typical packet path



Routing suites, bridge controllers, etc



Userspace Application
and
Vendor SDK

FDB/FIB

Init

Linux kernel structures

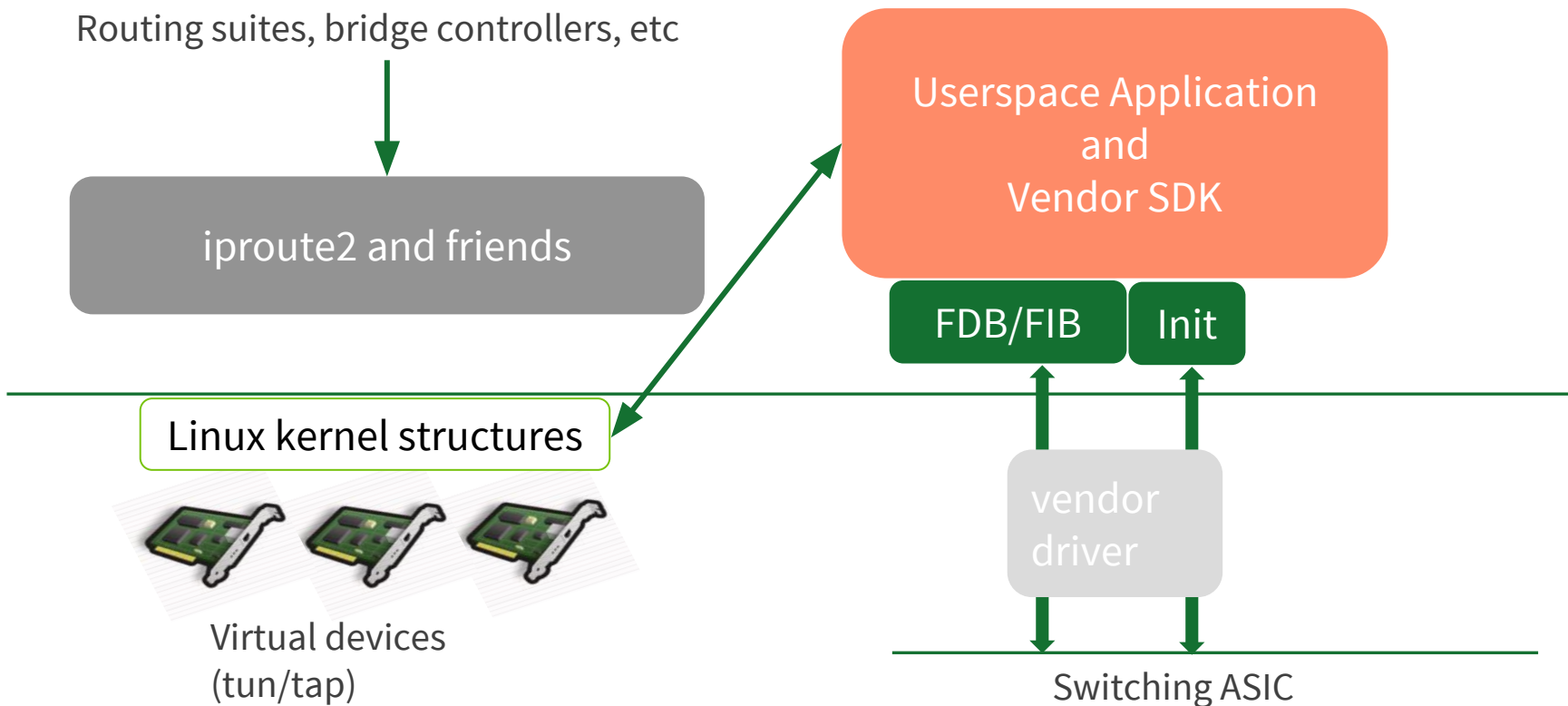


Virtual devices
(tun/tap)

vendor
driver

Switching ASIC

**Painful for those developing switches as
management applications need to talk to
kernel/netlink and SDKs**



**Much better design, but each SDK
supported needs a new translation
between netlink and SDK**

**Kernel hackers and distribution vendors
see a simple solution**

Get rid of all closed-source SDKs

Great idea!

**Vendors do not want to open
source their SDKs**

**Can we use a userspace SDK and a
kernel driver at the same time?**



Not if you want it upstream!

OK...how do I get started?

Phased Approach

- Participate!
- Pick a hardware platform
- Write and post a switchdev-compatible network driver
- Enhance that driver to add `ndo/offload_ops` to driver

**Attend conferences, participate on
mailing-lists, and post patches**

Write and post a switchdev-compatible network driver

Advantages

- Provide network access via front panel ports
- Phased approach to working upstream
- Applications can developed without need for hardware offload

What might that look like?

NIC tools and consumers

Socket
applications

Ethtool

Port Init

Link Mgmt

RxTx

Counters



switchdev compatible driver

Switching ASIC

Great, we are upstream!

Are we done?

No!

**Add offload support to driver as
upstream infrastructure is developed**

What might *that* look like?

NIC tools, Routing suites, bridge controllers, etc

Socket
applications

Ethtool

iproute2 and
friends

Port Init

Link Mgmt

Counters

RxTx

Offload Ops



switchdev compatible driver

Switching ASIC

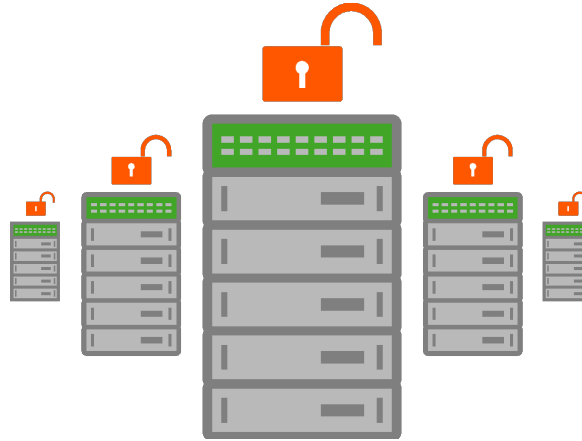
“If you are the first you will be so cool.”

-DaveM

Get coding

“..and we’ll help you maintain it”

-DaveM



Thank You!

© 2013 Cumulus Networks. Cumulus Networks, the Cumulus Networks Logo, and Cumulus Linux are trademarks or registered trademarks of Cumulus Networks, Inc. or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners. The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.