Stacked Vlan in Linux
- with Report from Netdev 0.1

Toshiaki Makita
NTT Open Source Software Center

Today's topics

- **Stacked vlan in Linux**
  - Stacked vlan overview, use-case and operation
  - Problems and approaches around stacked vlan
    - Offloading
    - MTU
- **Report from Netdev 0.1**
  - Summary of the conference
  - Hot topic: Offloading

Who is Toshiaki Makita?

- **Linux kernel engineer at NTT Open Source Software Center**
- **Technical support for NTT group companies**
- **Active patch submitter on kernel networking subsystem**
  - bridge, vlan, etc.

Topic 1

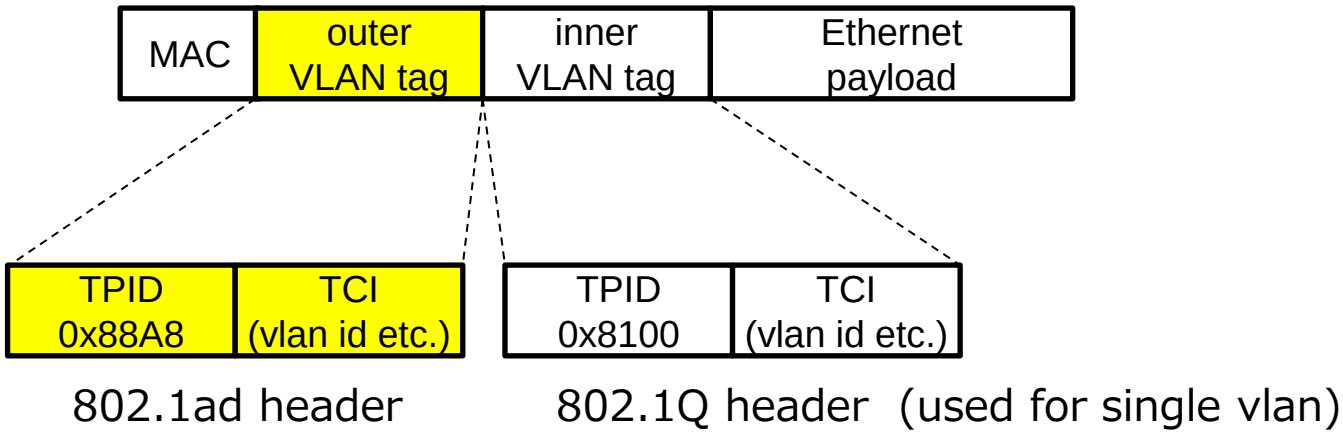
Innovative R&D by NTT

Stacked vlan in Linux

NTT

What is stacked vlan?

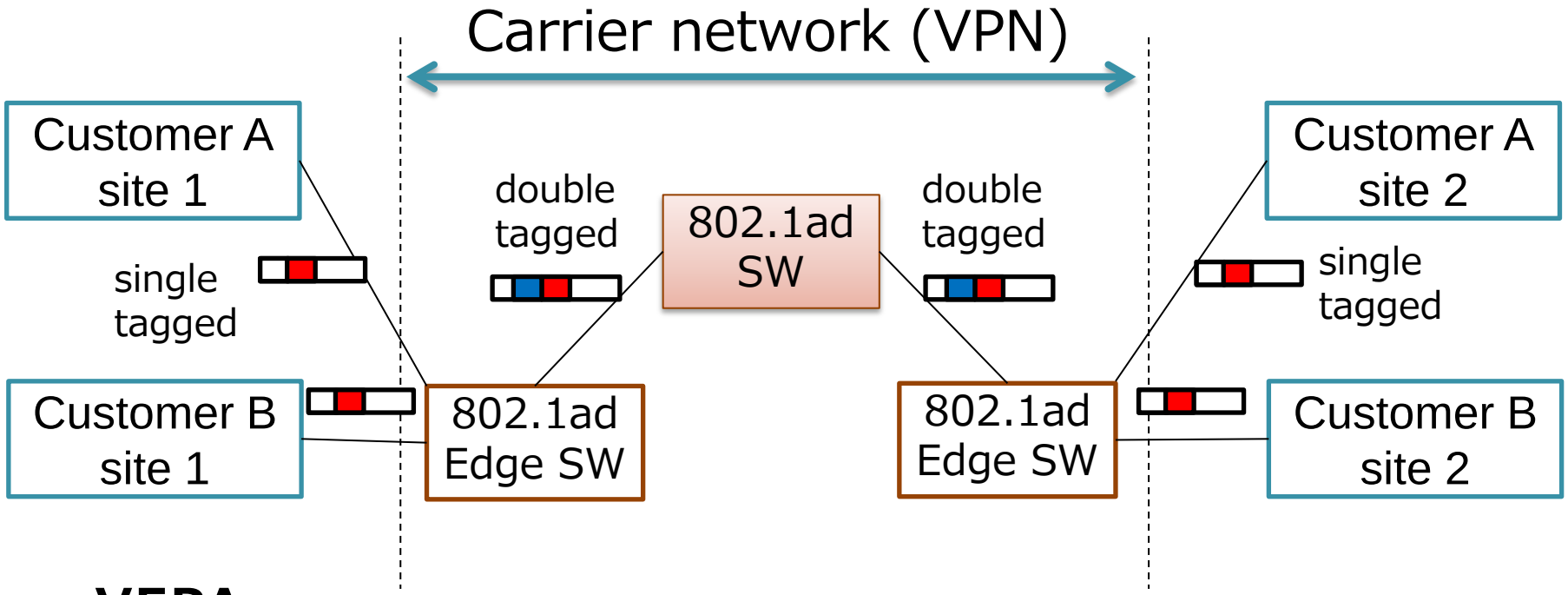
- **Stacked vlan:**
  - Two (or more) vlan tags in packets



  - Note: sometimes 802.1Q is also used for outer tag
    - Stacked vlan != 802.1ad

Where is stacked vlan used?

- **Ethernet VPN (Metro Ethernet)**
  - Outer tag is used to separate customers
  - Allow customers to use vlan (i.e. inner vlan) through VPN



- **VEPA**
  - Offload packet-forwarding between VMs to external switch
  - Use 802.1ad to separate VMs

How can we use stacked vlan on Linux?

- **Configuration examples for**
  - Outer = 802.1ad
  - Inner = 802.1Q

- **Case 1: Non-virtualization-host server**
  - Create 802.1ad vlan device
  - ... And create another vlan device on it

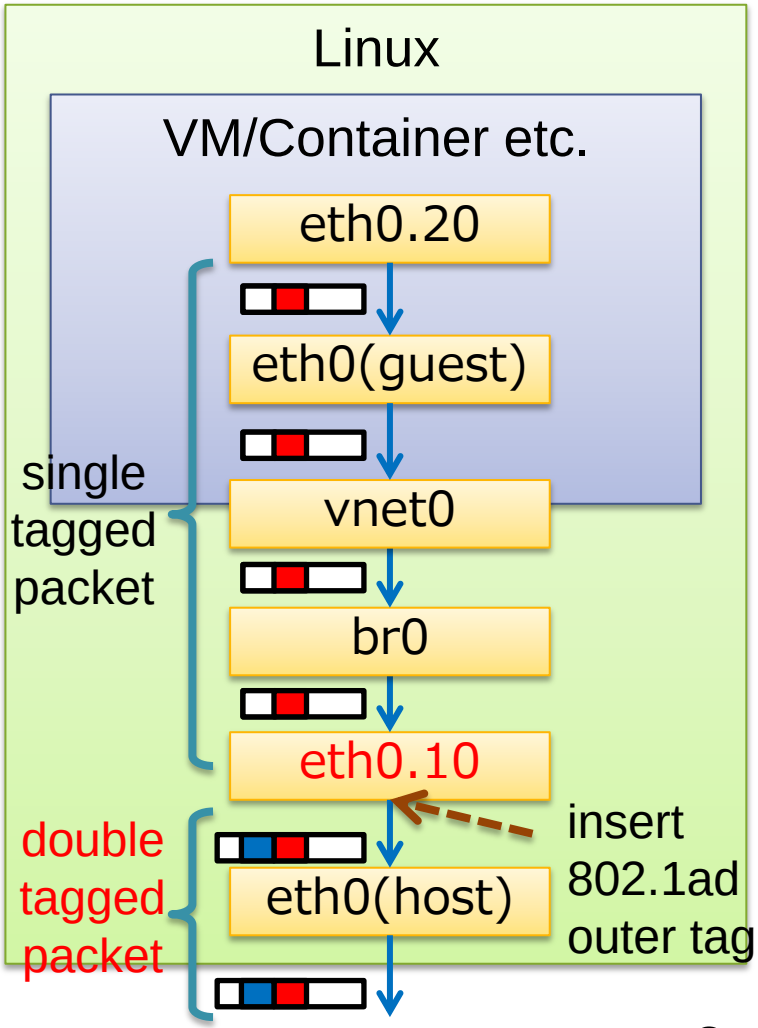
```
# ip link add link eth0 name eth0.10 type vlan id 10 protocol 802.1ad
# ip link add link eth0.10 name eth0.10.20 type vlan id 20
```

  - 802.1ad vlan device can be used since kernel 3.10

How can we use stacked vlan on Linux?

- **Case 2: VMs/containers host**
  - Assume VMs/containers use 802.1Q vlan
- **Case 2-a:**
  - Create 802.1ad vlan device on host

```
# ip link add link eth0 name eth0.10 type ¥
> vlan id 10 protocol 802.1ad
```

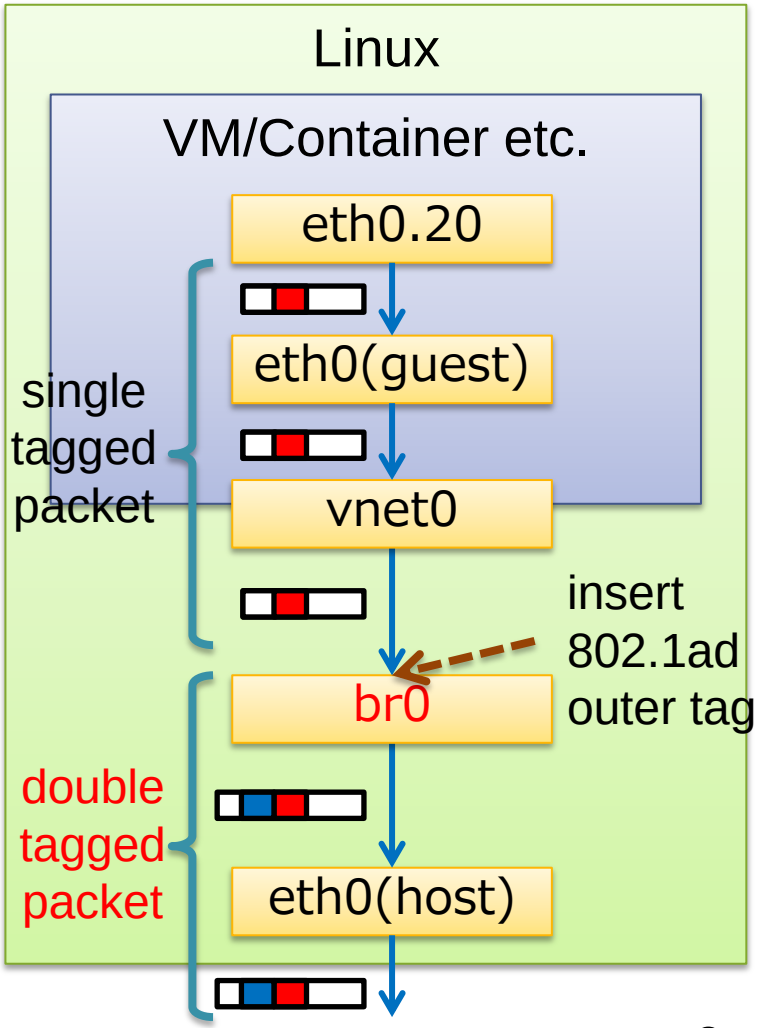


How can we use stacked vlan on Linux?

- **Case 2: VMs/containers host**
  - Assume VMs/containers use 802.1Q vlan
- **Case 2-b:**
  - Use bridge's vlan_filtering

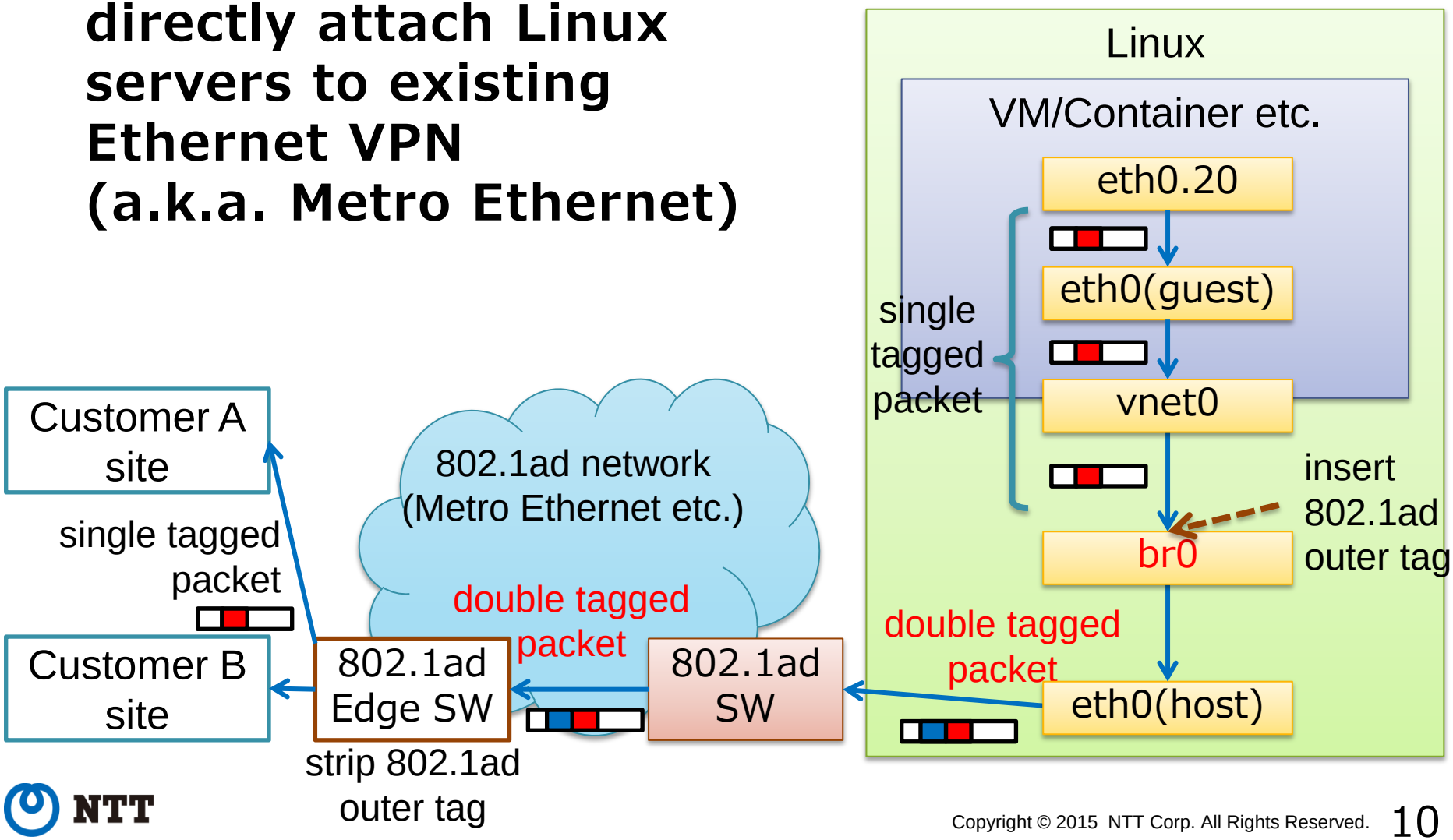
```
# echo 0x88a8 > ¥
> /sys/class/net/br0/bridge/vlan_protocol
# bridge vlan add dev vnet0 vid 10 pvid ¥
> untagged
# bridge vlan add dev eth0 vid 10
# echo 1 > ¥
> /sys/class/net/br0/bridge/vlan_filtering
```

  - 802.1ad vlan_protocol (0x88a8) can be used since kernel 3.16



Use case

- **802.1ad enables us to directly attach Linux servers to existing Ethernet VPN (a.k.a. Metro Ethernet)**



Topic 1-1

Innovative R&D by NTT

Problems around stacked vlan
1. Offloading

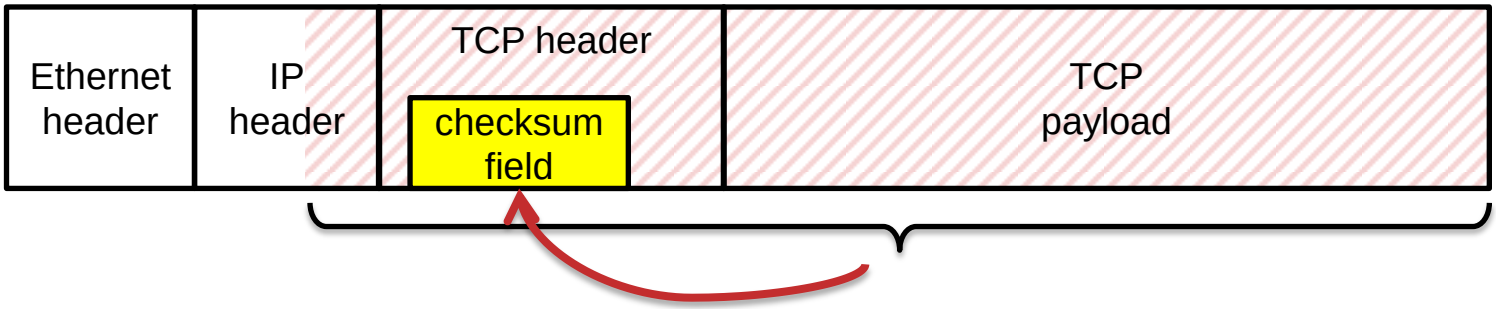
NTT

Offloading

- **Offloading**
  - Having NICs do some network task in place of CPUs
- **Tx side offloading features**
  - Checksumming, Segmentation, vlan-tag-insertion, ...
- **Rx side offloading features**
  - Checksumming, Large-receive, vlan-tag-parsing, ...

Tx side offloading - Checksum offload

- **Checksum offload**
  - Compute checksum field of L4 header (TCP, UDP, etc.)

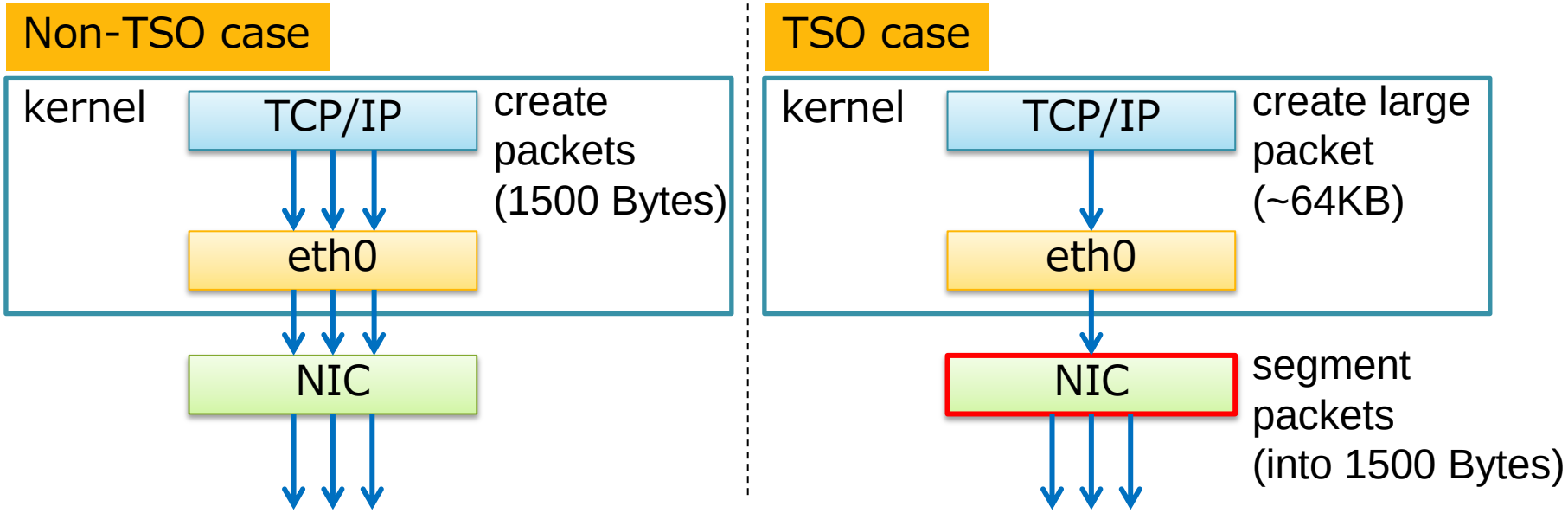


Whole tcp payload/header and some IP header fields are needed to compute checksum

  - Reduce calculation overhead/memory access on CPU

Tx side offloading - TSO

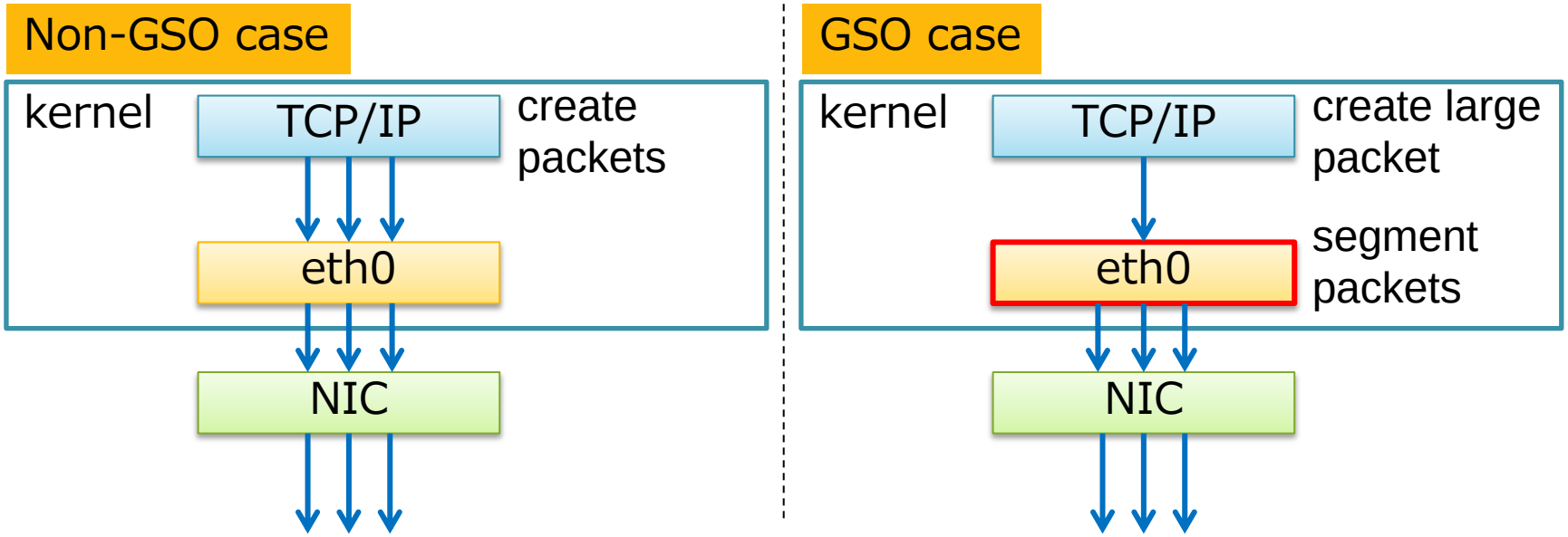
- **TCP segmentation offload (TSO)**
  - Split a large TCP packet into small (MTU-sized) packets



  - Reduce overhead in packet processing
  - Need checksum offload
    - Because NIC needs to calculate checksum for each segmented packet

Tx side offloading - GSO

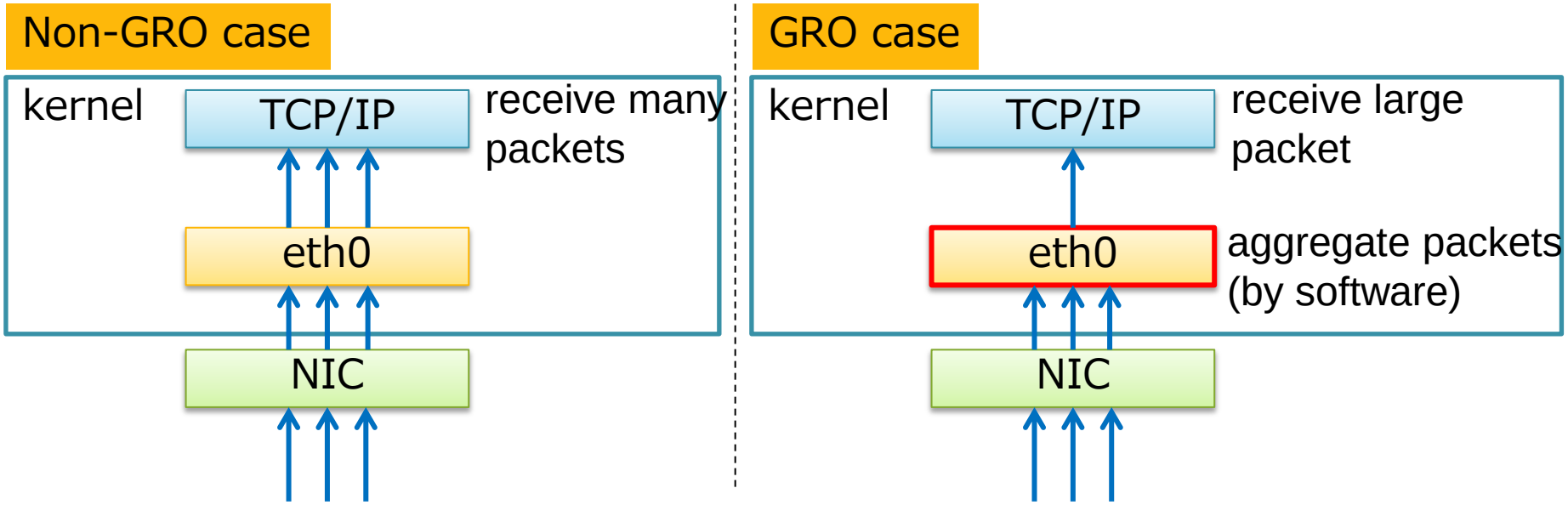
- **Generic segmentation offload (GSO)**
  - Split a large packet into small (MTU-sized) packets
  - Software emulation of TSO
    - can handle other protocols (UDP, GRE, VXLAN, ...)



- Reduce overhead in packet processing

Rx side offloading - GRO

- **Generic receive offload (GRO)**
  - Aggregate multiple packets into a large packet
  - Performed by software (offloading emulation like GSO)



  - Reduce overhead in packet processing

Offloading with stacked vlan

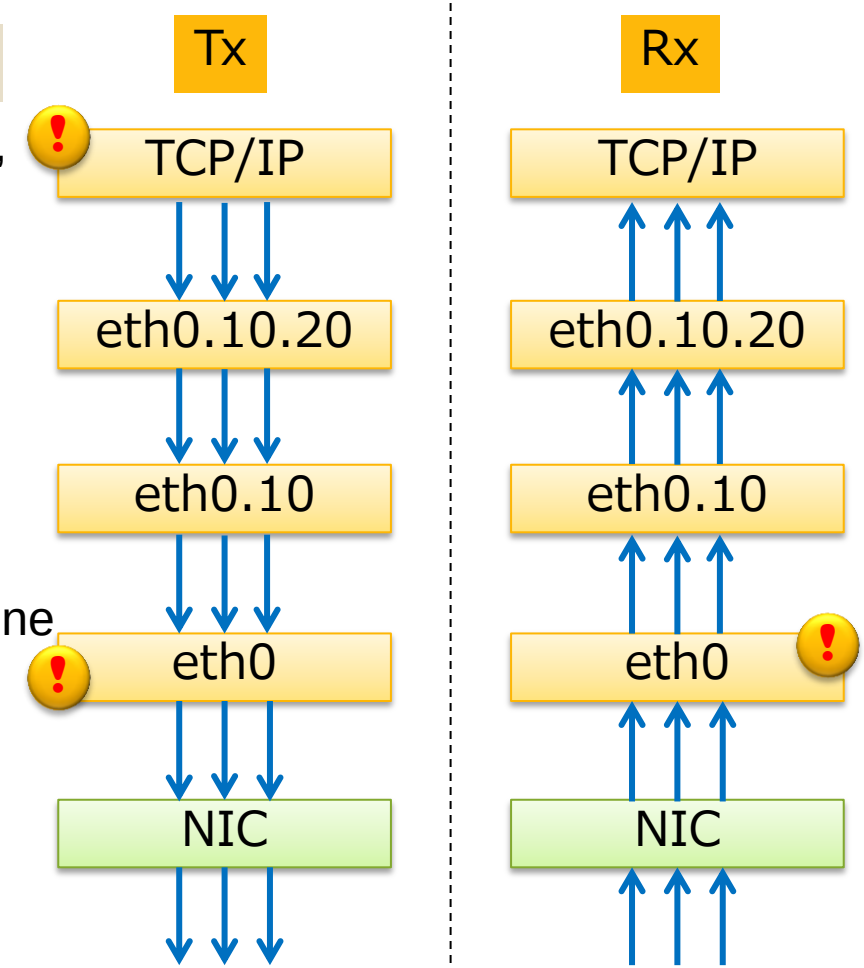
- ~kernel 4.0
  - Most offloading features get disabled with stacked vlan

no GSO/no checksum

create non-TSO packets,
compute checksum
by software

no TSO

no mechanism to determine
if NIC can segment
double tagged packets



no GRO

cannot aggregate
double tagged
packets

Offloading with stacked vlan - Being improved!

- **Future**
  - Major offloading features will be enabled with stacked vlan

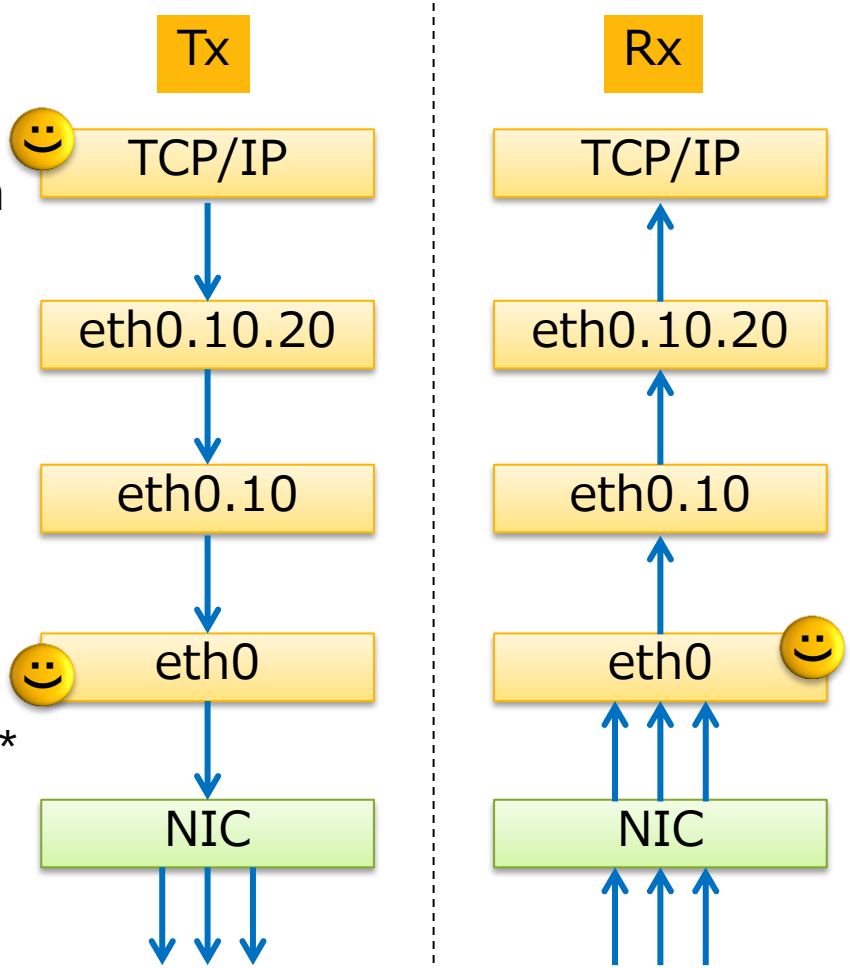
GSO/checksum

create TSO packets,
don't compute checksum
by software
(4.1-rc)

TSO

pass through double
tagged TSO packets
if NIC can segment them*
(4.1-rc)

* Currently only igb, bonding, and team have this feature

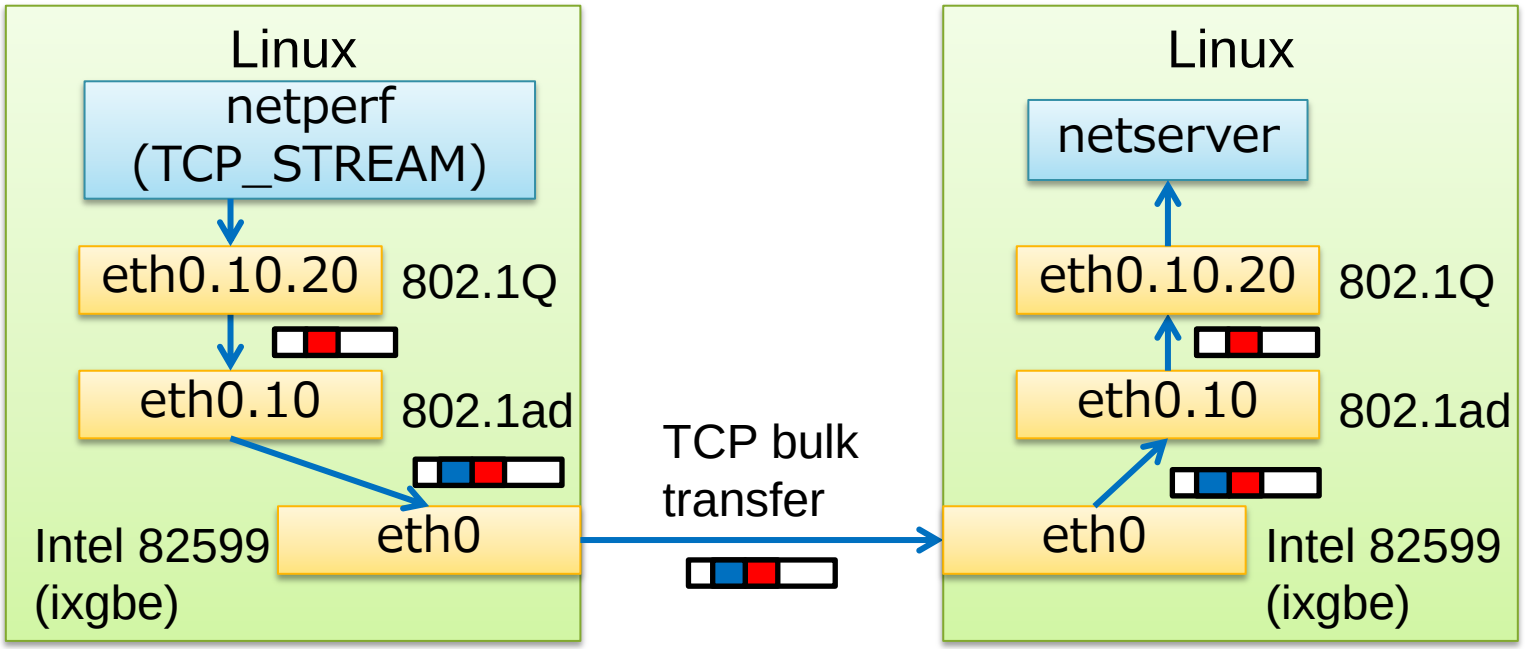


GRO

aggregate
double tagged
packets
(4.2?)

Performance test: Environment

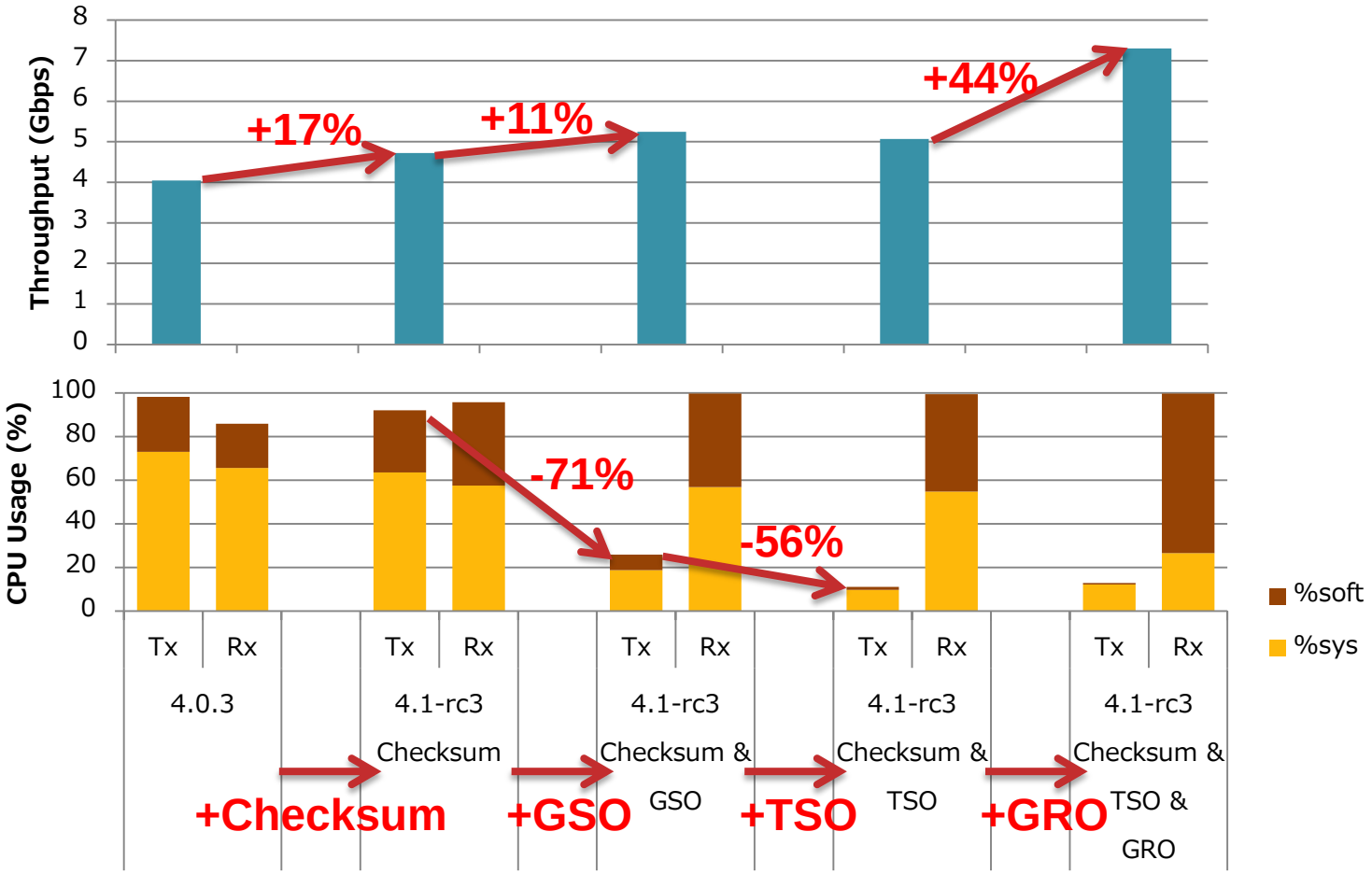
- **kernel**
  - 4.0.3 (no checksum/GSO/TSO/GRO)
  - 4.1-rc3 (with only checksum enabled)
  - 4.1-rc3 (with checksum/GSO enabled)
  - 4.1-rc3 (with checksum/TSO enabled)*1
  - 4.1-rc3 (with checksum/TSO/GRO enabled)*2
- **CPU: Xeon E5-2407 * 1core**
- **NIC: Intel 82599 (ixgbe)**
- **Benchmark tool: netperf-2.6 (TCP_STREAM)**



*1 Apply a patch to pass through double tagged TSO on ixgbe, like igb
*2 Apply a patch to enable double tagged GRO from net-next tree

Performance test: Result

- Checksum/GSO/TSO/GRO drastically improve throughput and overhead

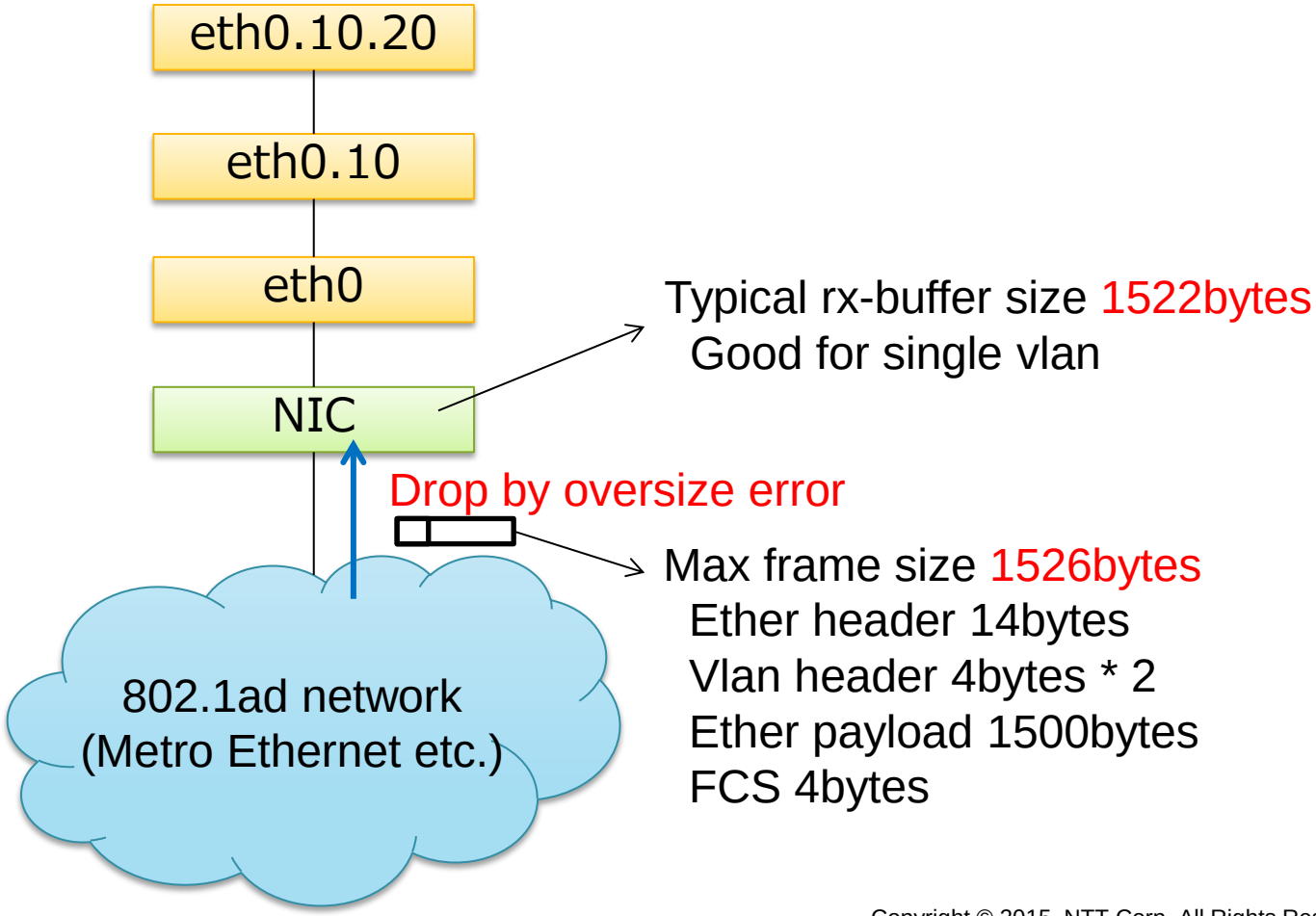


Innovative R&D by NTT

Problems around stacked vlan
2. MTU

MTU problem

- **MTU-sized double tagged packets are dropped by default due to oversize error**



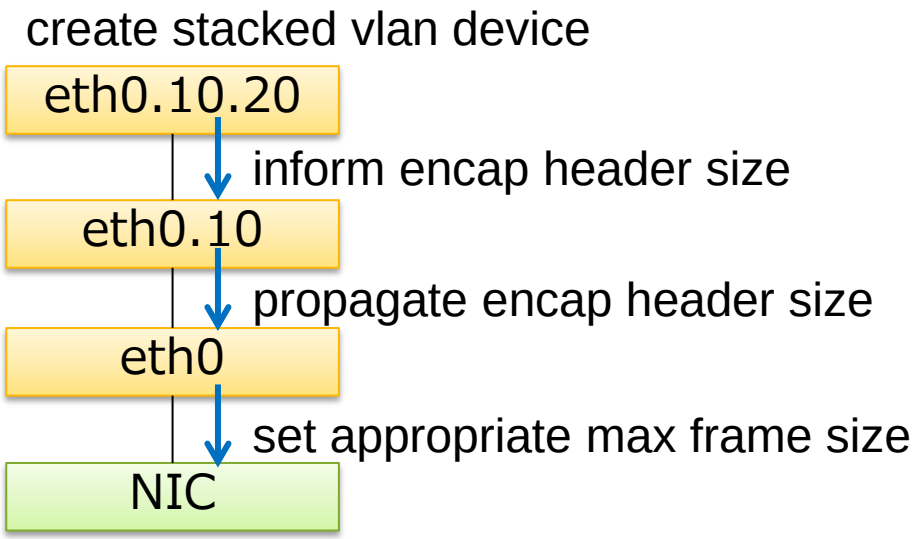
What's so problematic?

- **Looks like a strange random failure**
  - Ping is OK
  - TCP connection can be established
  - SSH is mostly OK
  - Only large packets are discarded
  - Hard to identify the root cause for admins

- **Workaround differs from driver to driver**
  - Setting MTU to 1504 should work in most cases
  - Sometimes 1508/9000 is needed depending on drivers

Approach

- **Automatically adjust buffer size on creating stacked vlan device**
  - Introduce a new driver-API to inform the size of encapsulation header
  - Need implementation in every driver
  - Could be used for other encapsulation protocol (mpls, vxlan, etc.)
  - Under development..



Topic 2

Innovative R&D by NTT

Report from Netdev 0.1

NTT

Netdev 0.1

Innovative R&D by NTT

- 150 of Linux netheads got together first

Key note: Photo by Richard Guy Briggs, Licensed under CC BY-NC-SA 2.5 CA

NTT

What is Netdev 0.1?

- **Netdev 0.1 (year 0, conference 1) is a community-driven conference focusing on Linux networking**
  - Feb. 14 - 17, 2015
  - @Ottawa, Canada
- **Talks, Workshops, BoFs, and Tutorials**
- **Held for first time**
- **Everyone who registered can join (not invitation-only event)**
- **-20C, -4F**
- **Stacked vlan problems shown here were discussed in BoF as well**

Conference venue: Photo by me :)

Netdev topics

- **Netdev greatly covers Linux networking topics**
  - Offloading
  - Performance analysis/improvement
  - User space networking
  - New protocols
  - Container networking
  - Overlay networking/tunneling
  - Netfilter/Nftables
  - Traffic control

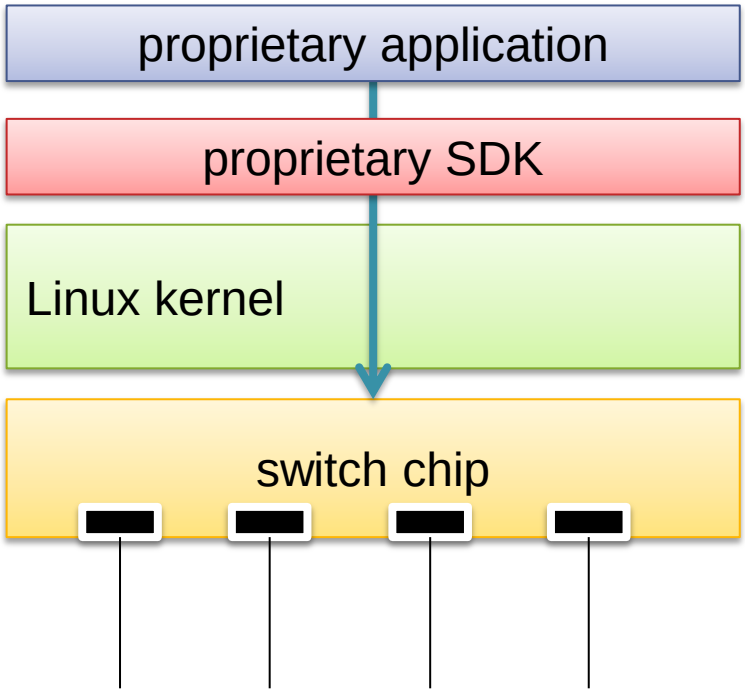
- **Offloading was especially featured this year**

Hardware Offloading BoF: Photo by Richard Guy Briggs
Licensed under CC BY-NC-SA 2.5 CA

Offloading for bridge, etc

- Quite a few talks (30-40%) in Netdev are related to this topic

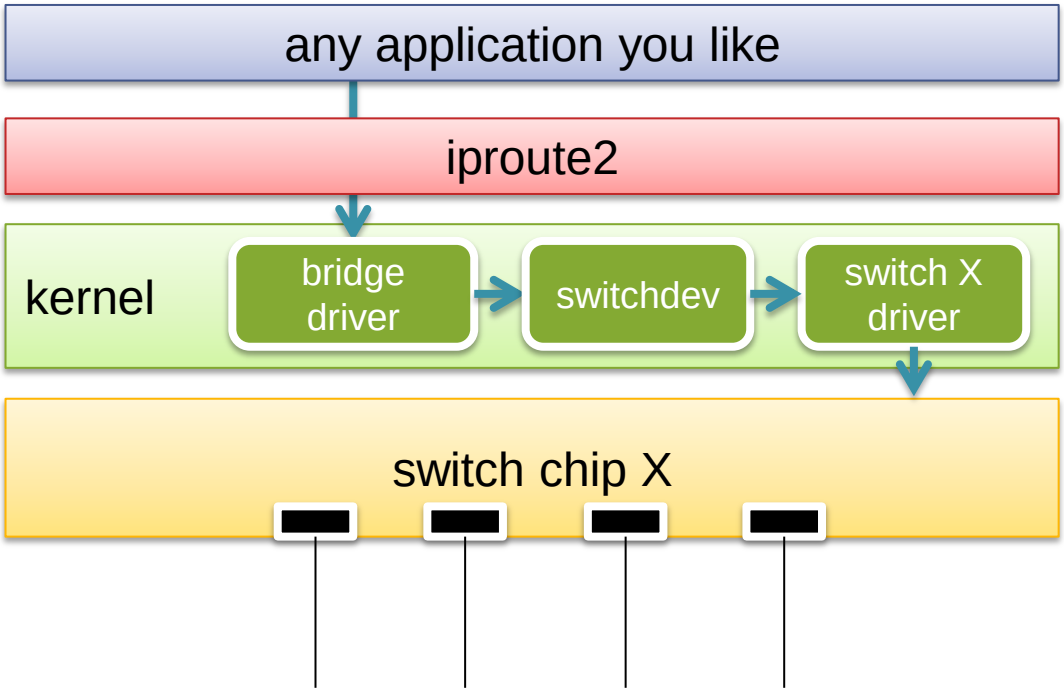
- Some HW switches use Linux for their OS
- But...



Offloading for bridge, etc

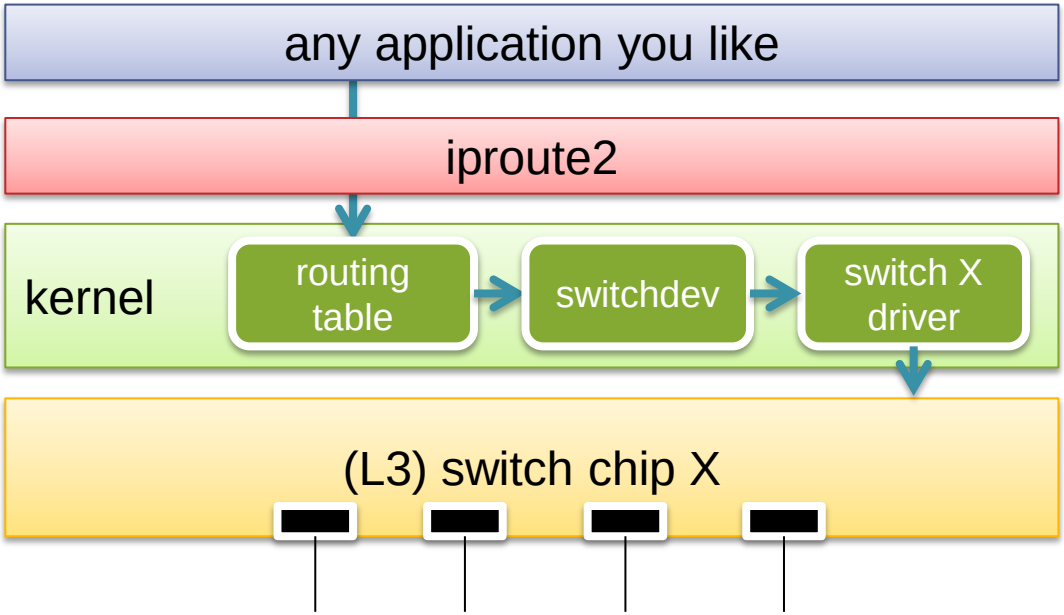
- Linux introduced a new model "switchdev" (kernel 3.19)

- Hardware switches can be used in the same way as software bridges



Offloading for bridge, etc

- **Similarly, routing can be offloaded (kernel 4.1-rc)**



- **Offloading of ACL and flow-based networking have been under discussion...**
  - Multiple different ways to offload have been proposed

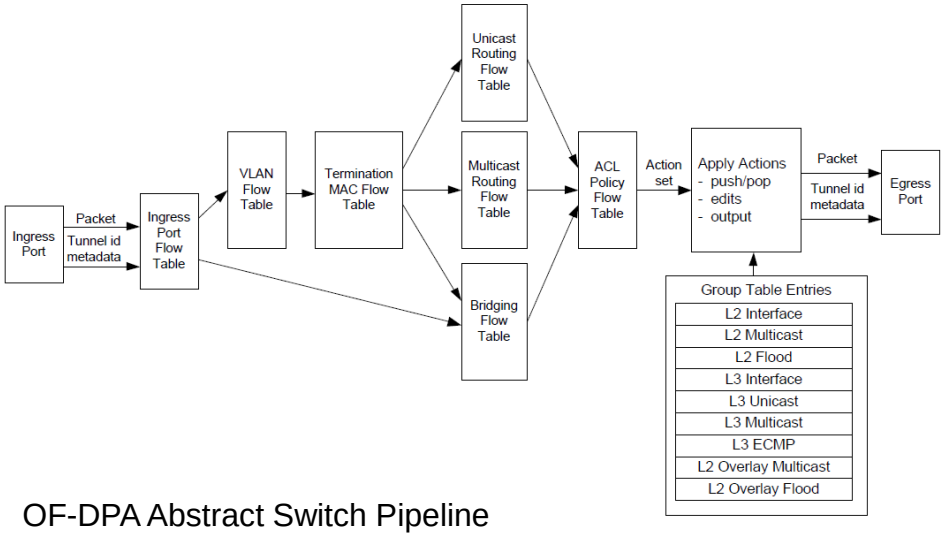
Use cases

- **White-box switches**
  - Linux for switch OS
  - Able to run any apps you like on switches

- **NIC-embedded switches**
  - Some NICs have embedded switches in them
  - We could utilize switch functionality on servers

Rocker

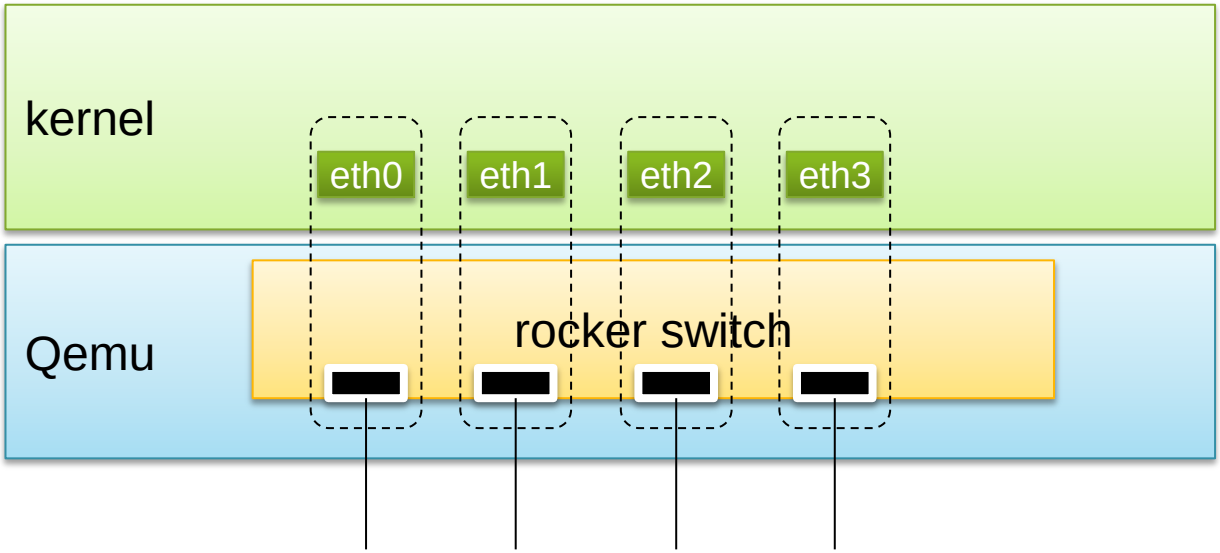
- **You can try bridge/routing offload by rocker**
- **rocker**
  - Virtualized hardware switch implemented in Qemu
  - Created for testing and prototyping purpose
  - Supports switching and routing
  - Based on Broadcom's OF-DPA (OpenFlow Data Plane Abstraction) model



OF-DPA Abstract Switch Pipeline
http://www.broadcom.com/collateral/etp/OFDPA_OASS-ETP101-R.pdf

Bridge/routing offload behavior

- **Each switch (router) port is exposed as an ethernet device (eth0, eth1, ...)***
- **By default, each port behaves as a standalone router port**

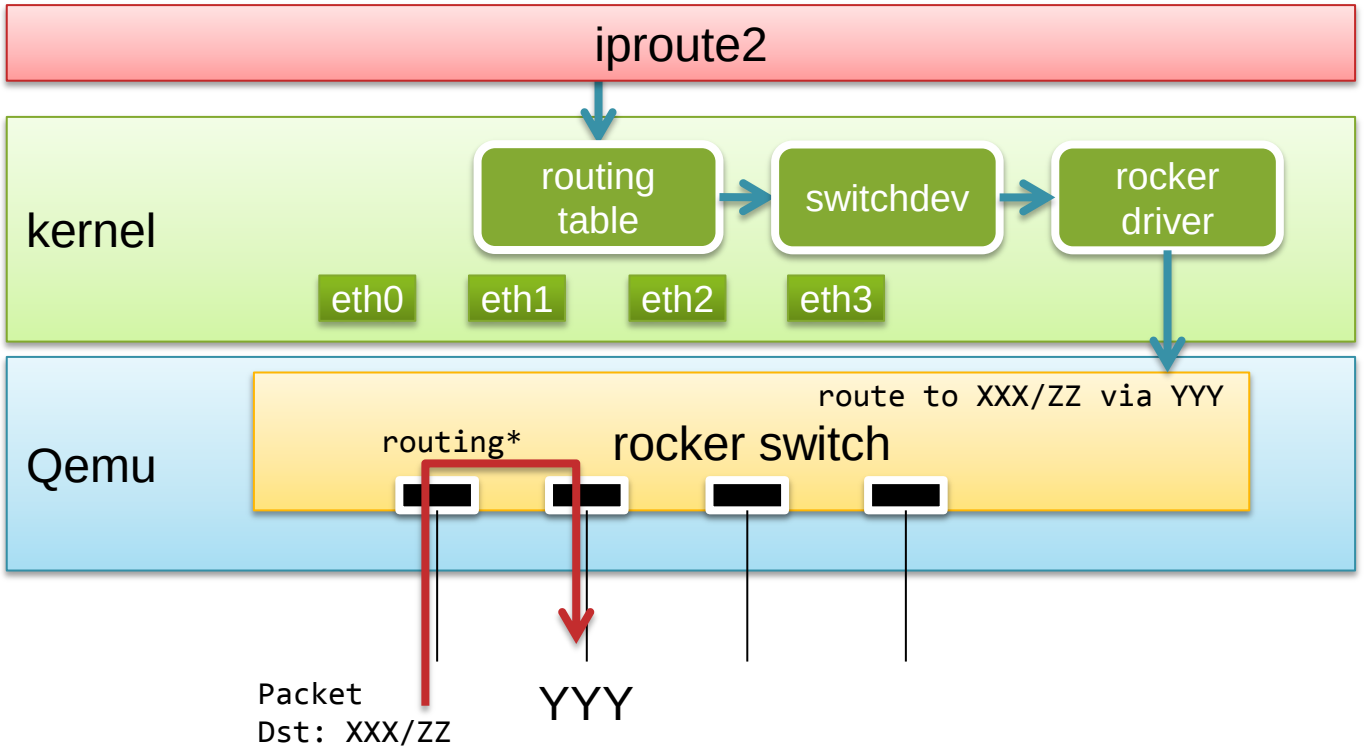


* Although suggested naming convention is "swXpYsZ", "ethX" is used in this slides to explain that they are exposed like normal ethernet devices.

Routing

- **Routes added to routing table of Linux are automatically offloaded**

- add route `# ip route add XXX/ZZ via YYY`

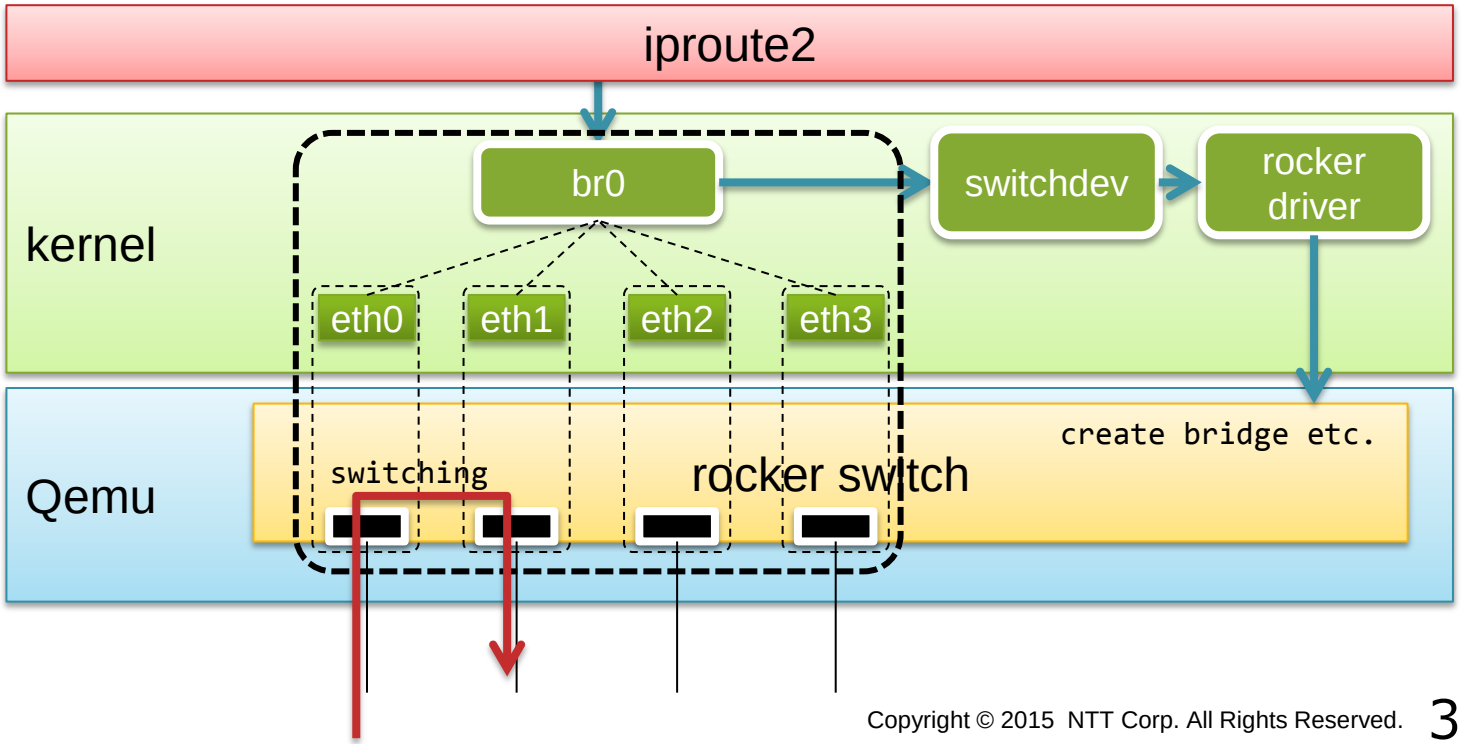


* Routing in the switch can be done after neighbour (ARP, etc.) entry to its nexthop is resolved by kernel

Bridge

- **Hardware bridges can be created as if software bridges**

- create bridge `# ip link add br0 type bridge (brctl addbr br0)`
- attach port to bridge `# ip link set eth0 master br0 (brctl addif br0 eth0)`
- assign vlan `# bridge vlan add dev eth0 vid 10`



Hardware offloading future

- **Challenges still under discussion...**
  - Driver for a real switch chip!
  - ACL (netfilter/nftables) offloading
  - Flow-based networking (OpenFlow, etc.) offloading
  - Same model for SR-IOV
    - Most SR-IOV NICs can forward VM-VM/VM-host traffic through NIC embedded switch
    - Currently Linux provides different API to manipulate SR-IOV features

Summary

- **Topic 1: Stacked vlan in Linux**
  - Two ways to use stacked vlan
    - vlan device
    - bridge's vlan_filtering
  - Performance problem on offloading
    - being improved
  - MTU problem
    - still not resolved, but you can try setting MTU by hand

- **Topic 2: Netdev 0.1**
  - Offloading was hot this year
  - Great fit for people who love networking
  - Anybody can register; let's join next year!
    - will happen in Sevilla, Spain, Q1 2016 (maybe warm..)

Thank you!