Design of Vhost-pci

- designing a new virtio device for inter-VM communication

Wei Wang
wei.w.wang@intel.com

Contributors:
Jun Nakajima, Mesut Ergin, James Tsai, Guangrong Xiao,
Mallesh Koujalagi, Huawei Xie, Yuanhan Liu

Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Agenda

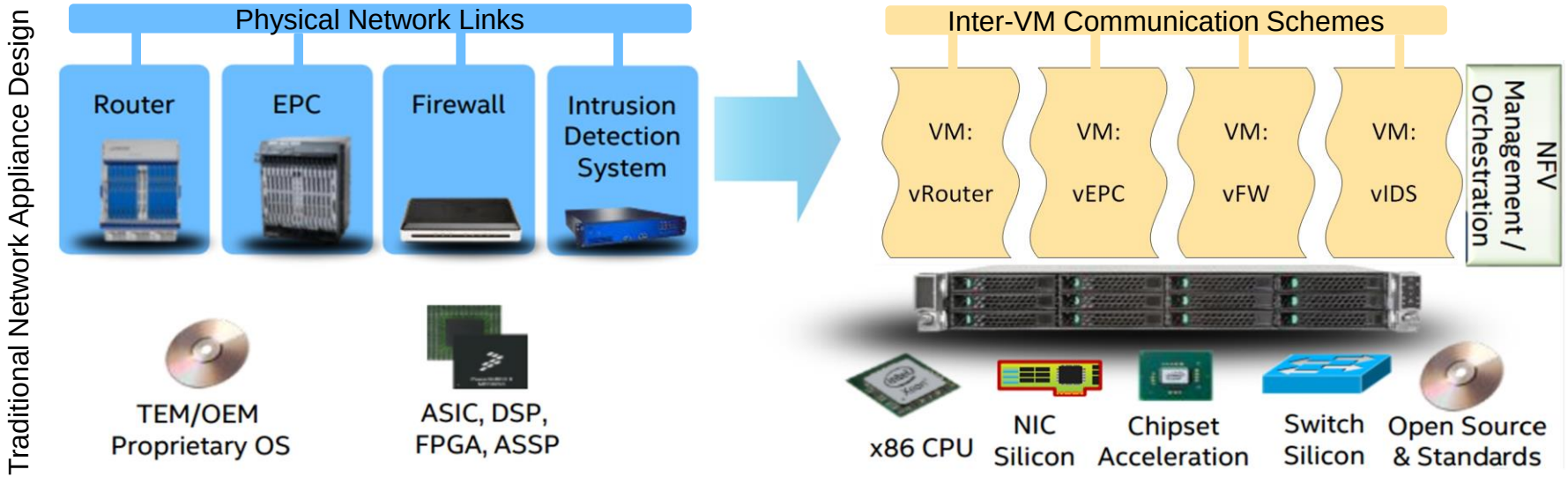
Part 1: Usage and Motivation

Part 2: Design Details

Part 3: Current Status

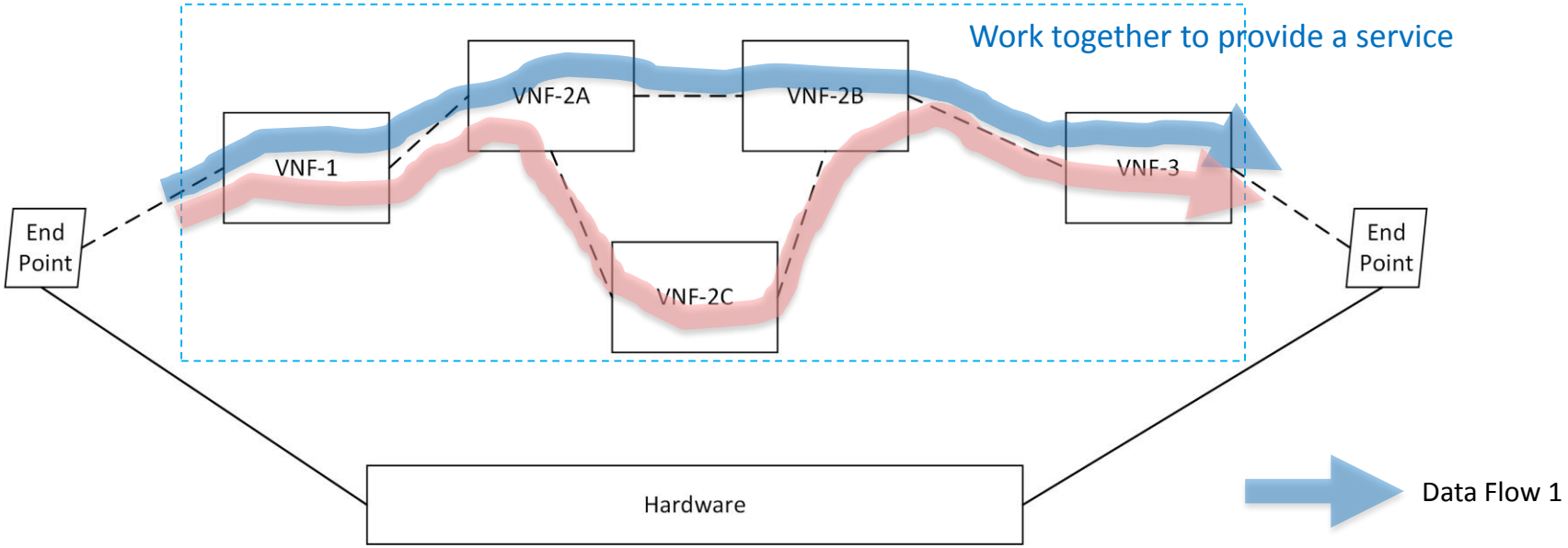
Part 1:
Usage and Motivation

Transformation of Network Appliances



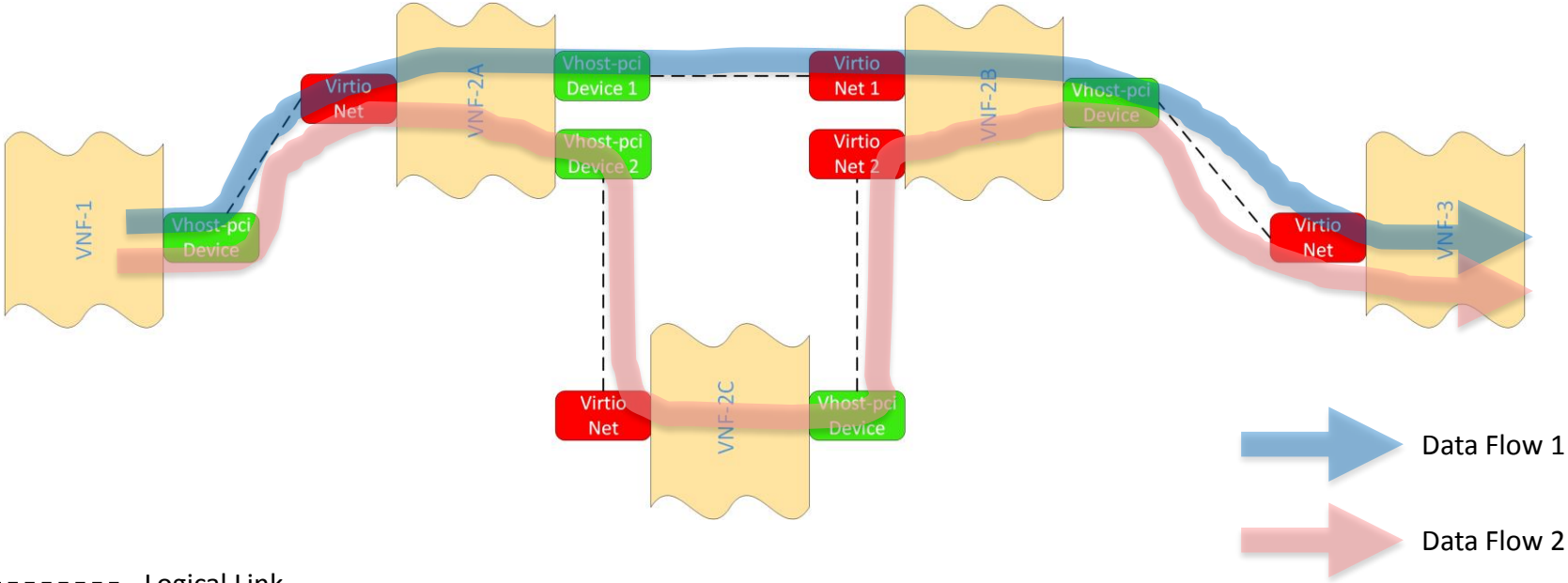
Network Appliances to Virtual Network Functions(VNF):
transformation relies on high performance inter-VM communication schemes

Virtual Network Function Forwarding Graph



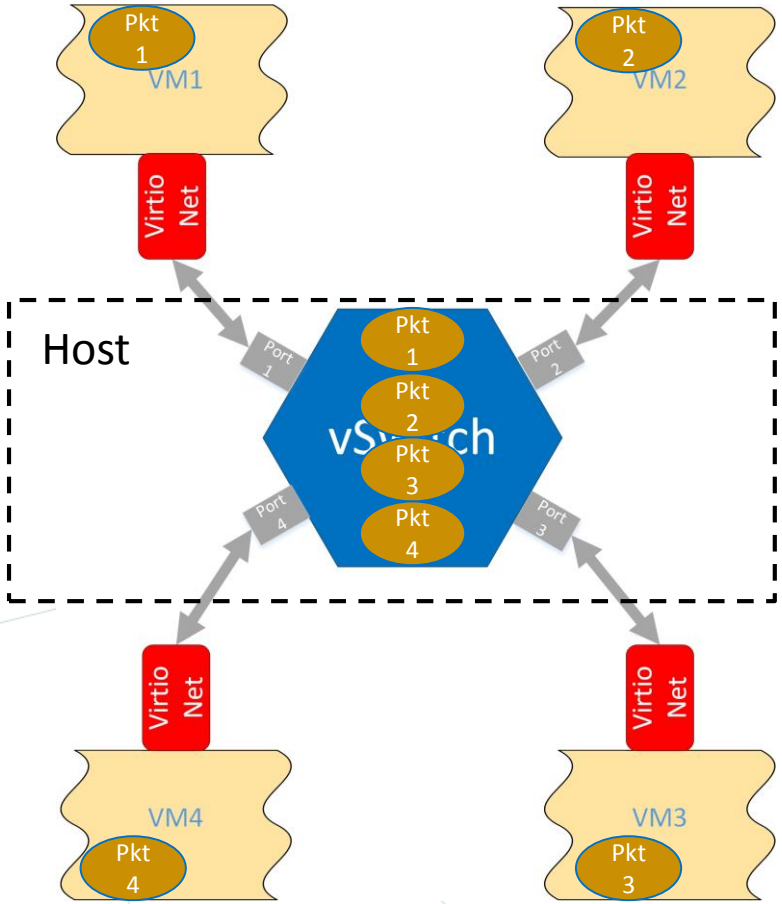
Ref: ETSI, "Architectural Framework", 2013
http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf

VNF Forwarding with Vhost-pci



Existing Inter-VM Network Packet Transmission

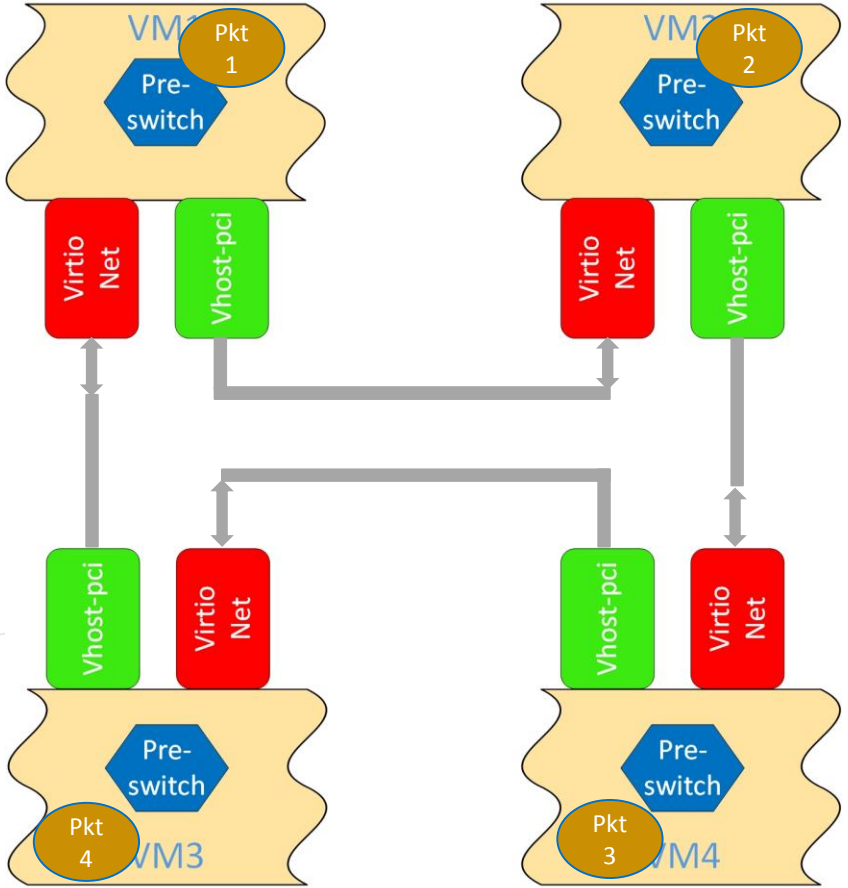
- Long Code Path: packets are transmitted from one VM to another via an intermediary
- Packets, streamed out of VMs, are bumper-to-bumper in the central vSwitch



Vhost-pci for Inter-VM Network Packet Transmission

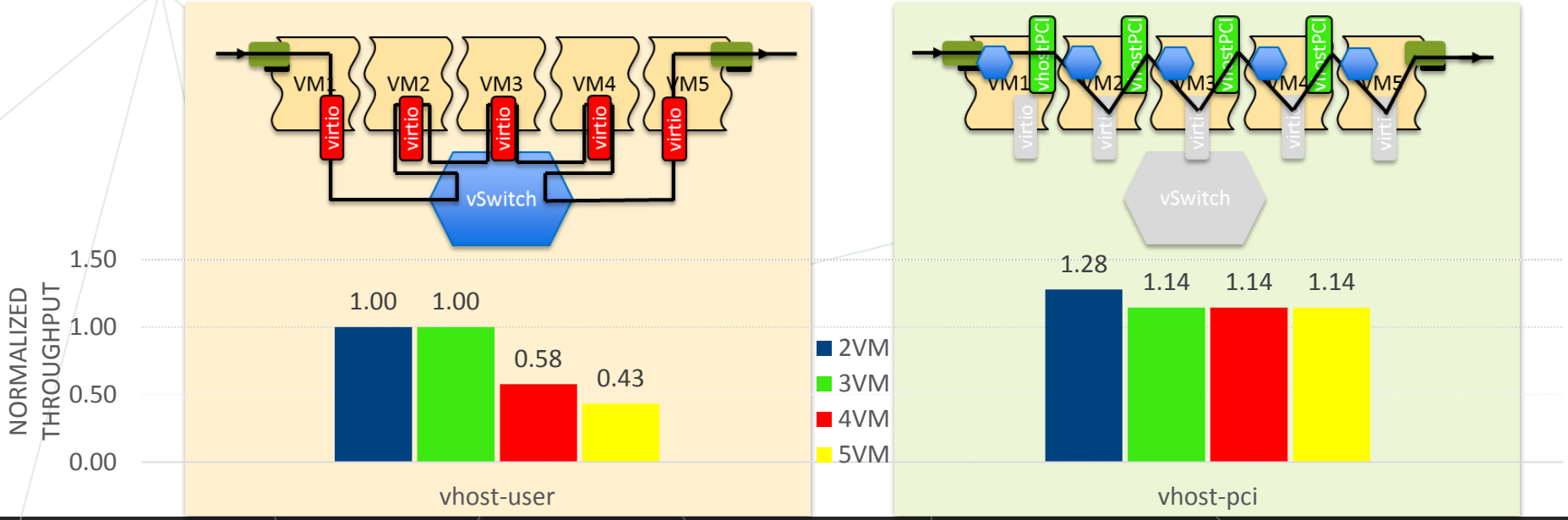
Advantages:

- Short Code Path: packets are transmitted from one VM directly to another VM
- Better scalability



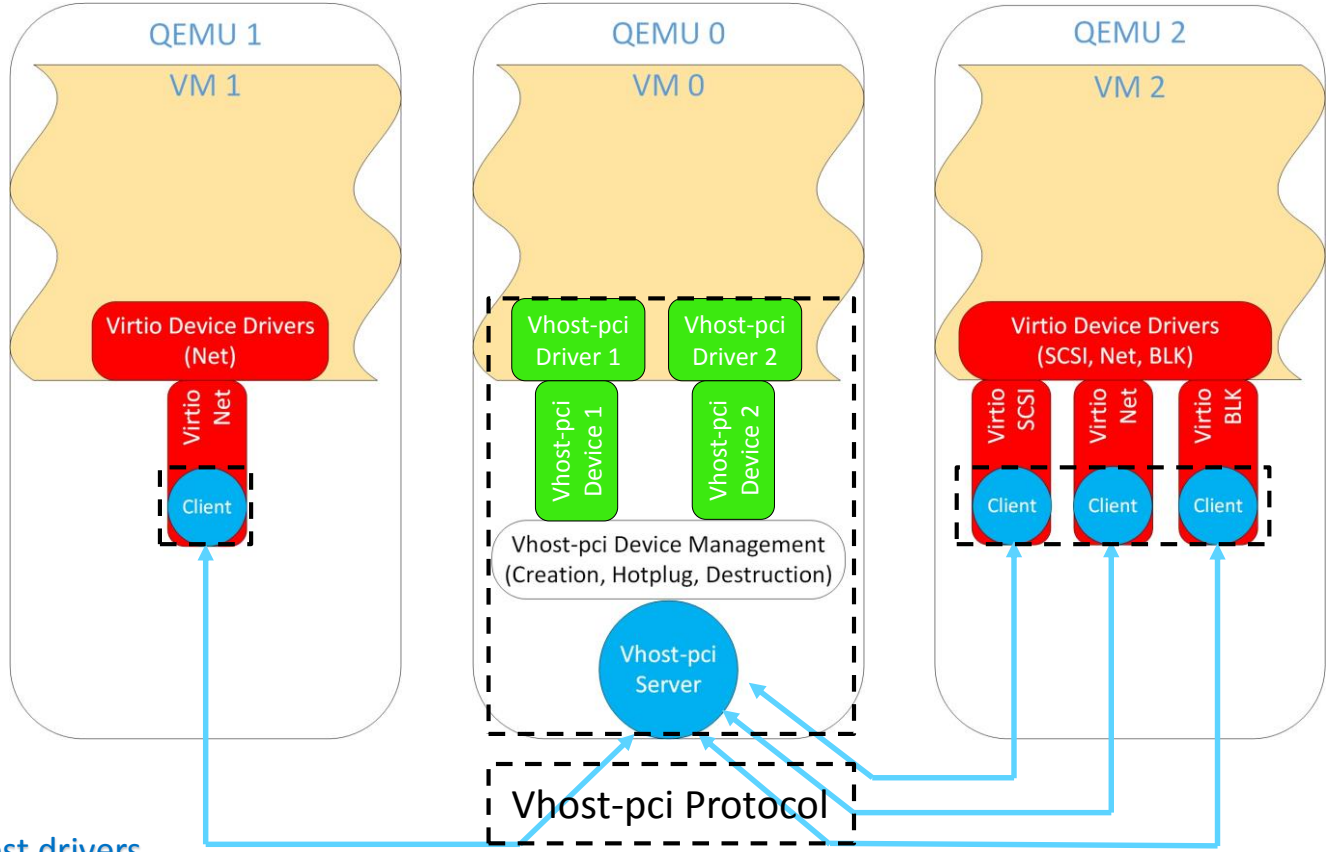
Micro-benchmarking Results

VSPERF / Chain of 2 to 5 VM
- RFC2544 via ext. packet generator DPDK Pktgen
- OVS DPDK on two cores (default)
- VM setup: one pinned vCPU, 2GB RAM (hugepages)
- pCPU: Intel(R) Xeon(R) E5-2698 v3 @ 2.30GHz



Part 2:
Design Details

Vhost-pci Design



Frontend Device/Driver

Backend Device/Driver

QEMU Socket Server/Client

Socket Connection

New Component

No change needed to in-guest drivers for virtio devices

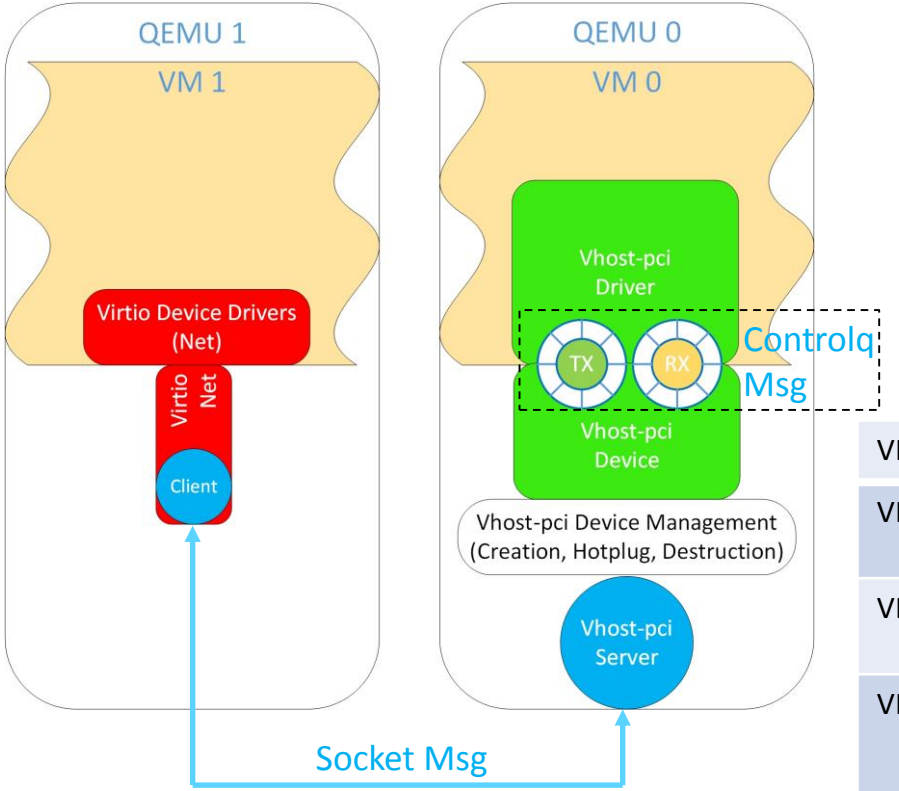
Vhost-pci Server

- To use the vhost-pci based inter-VM communication mechanism, a VM's QEMU needs to create a vhost-pci server
- Creates a vhost-pci-server by adding the following QEMU booting commands:
  - -chardev socket,id=vhost-pci-server-xyz,server,wait=off,connections=32,path=/opt/vhost-pci-server-xyz
  - -vhost-pci-server socket,chardev=vhost-pci-server-xyz

Vhost-pci Client

- To use a vhost-pci device on another VM as a backend, the originating virtio device supplies a vhost-pci client which connects to the remote vhost-pci server
- Create a virtio device with a vhost-pci client using the following commands:
  - -chardev socket,id=vp-client1,path=/opt/vhost-pci-server-xyz
  - -device virtio-net-pci,mac=52:54:00:00:00:01,vhost-pci-client=vp-client1
- The client communicates to the server using the vhost-pci protocol to set up the inter-VM communication channel

Vhost-PCI Protocol

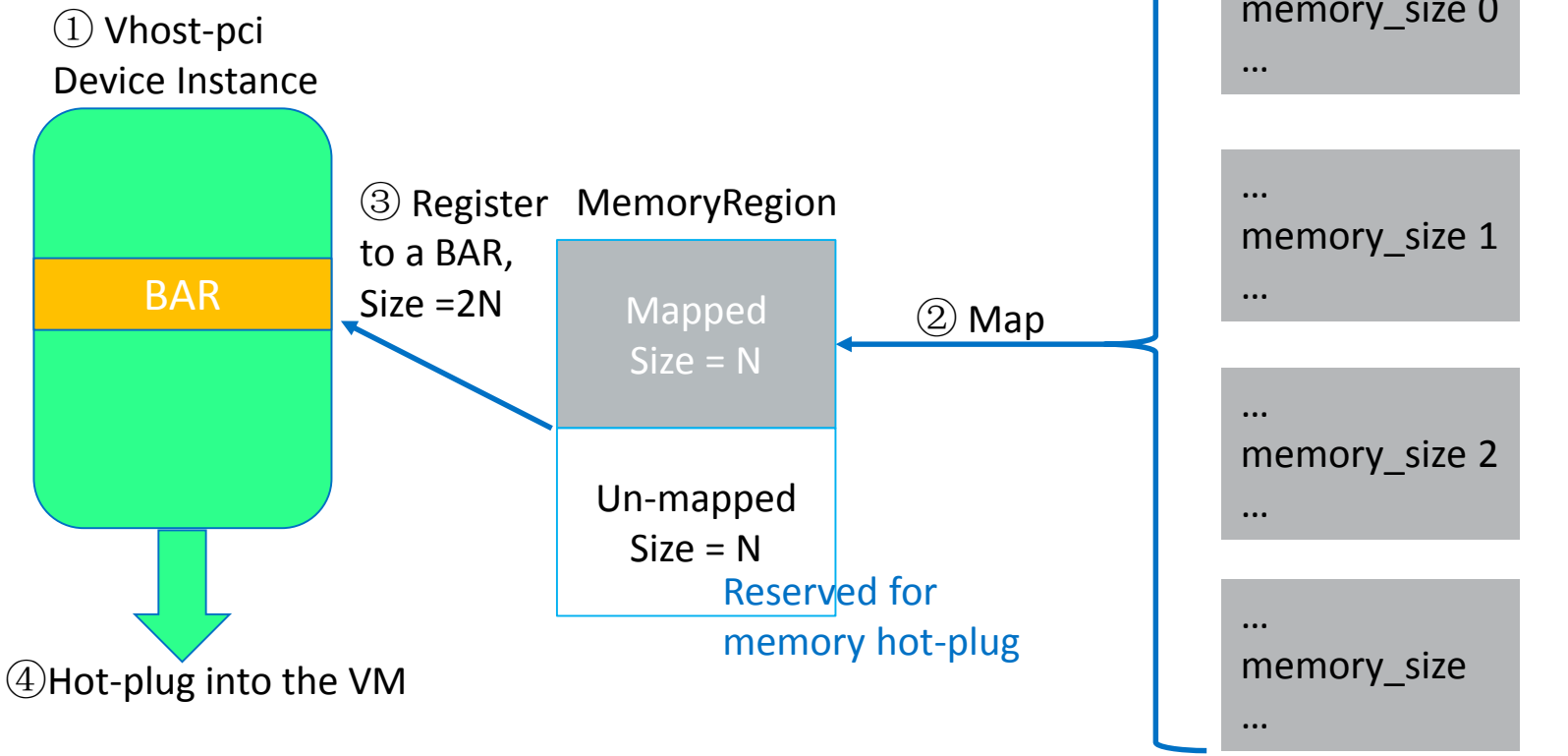


Protocol Msg

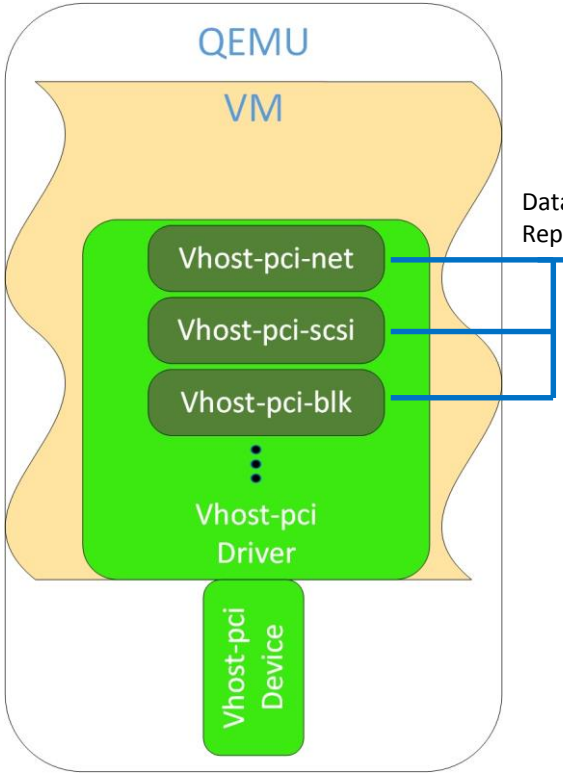
VHOST_PCI_GET_UUID	Identifies a frontend VM
VHOST_PCI_GET_MEMORY_INFO	Used to map the entire frontend VM's memory
VHOST_PCI_GET_DEVICE_INFO	Frontend device info (device type, vring addr etc)
VHOST_PCI_GET_FEATURE_BITS	Feature bits of the frontend device to be negotiated with the vhost-pci device and driver

Vhost-pci Device Management

Device Creation



Vhost-pci Driver



Data Structure Representation

struct vhost_pci_info:

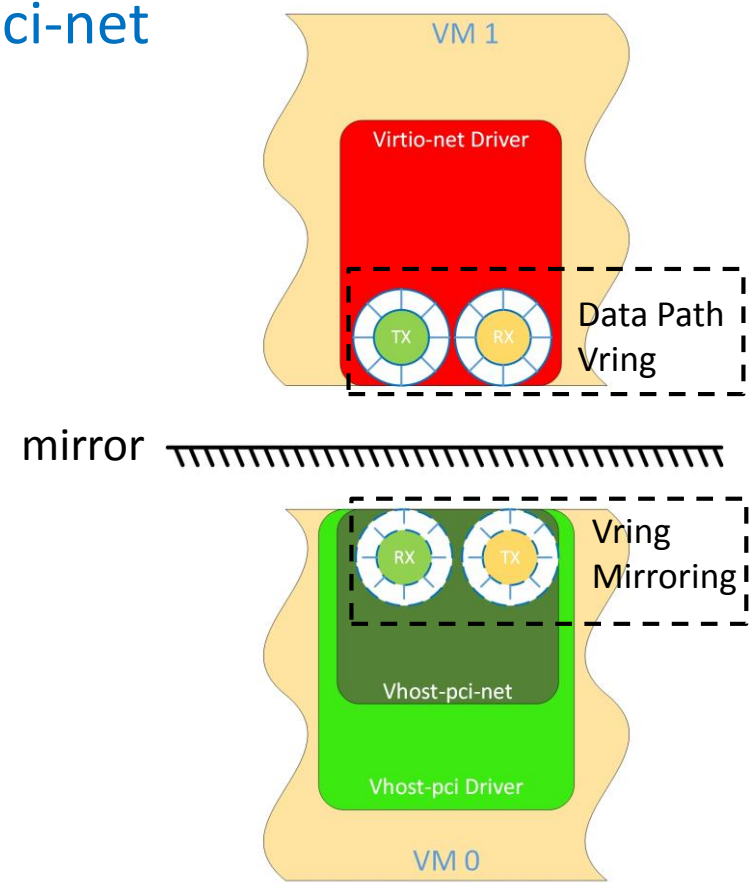
- struct vhost_pci_dev[MAX_NUM];

struct vhost_pci_dev:

- u32 device_type;
- u64 device_id;
- void *dev;

Pointer to the device specific structure
e.g. dev = net_device

Vhost-pci-net



- vhost-pci-net shares vrings created by the originating virtio-net device
- TX ring from originating device becomes RX ring at mirrored device, and vice versa
- Copying packets in and out of originating device rings is the responsibility of vhost-pci-net

Part 3:
Current Status

Current Status

- Initial PoC completed, summary of results presented
- Design RFC v2 has been sent out to KVM/QEMU mailing list (https://lists.gnu.org/archive/html/qemu-devel/2016-06/msg05359.html)
- Patches implementing RFC v2 design are work in progress

End of Presentation

Thank you!