Introduction to Oracle VM (Xen) Networking

Dongli Zhang

Oracle Asia Research and Development Centers (Beijing)

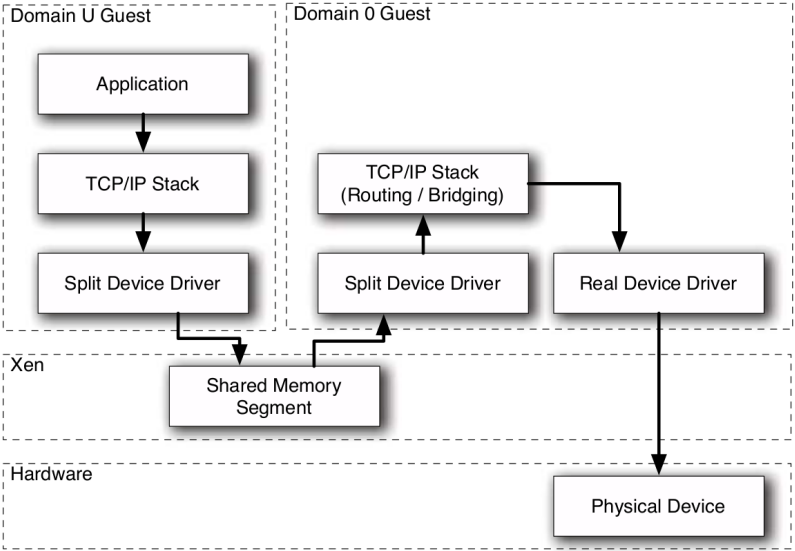
dongli.zhang@oracle.com

March 8, 2017

Plan

- Paravirtualized Networking
  - vif, bridge, bond
- Emulated Networking
- Environment:
  - xen: Oracle VM server 3.3.3 with xen-4.3.0-55.el6.47.33.x86_64
  - dom0: Unbreakable Enterprise Kernel v4.1.12-89
  - domU: Unbreakable Enterprise Kernel v4.1.12-89
- Prerequisite Knowledge: http://finallyjustice.github.io/xen-arch.pdf
  - xen framework
  - PVM vs. HVM vs. PVHVM
  - event channel, grant table
  - xen admin hands-on experience (preferred)

Paravirtual xen-netfront/xen-netback framework



xen-netfront/xen-netback source code

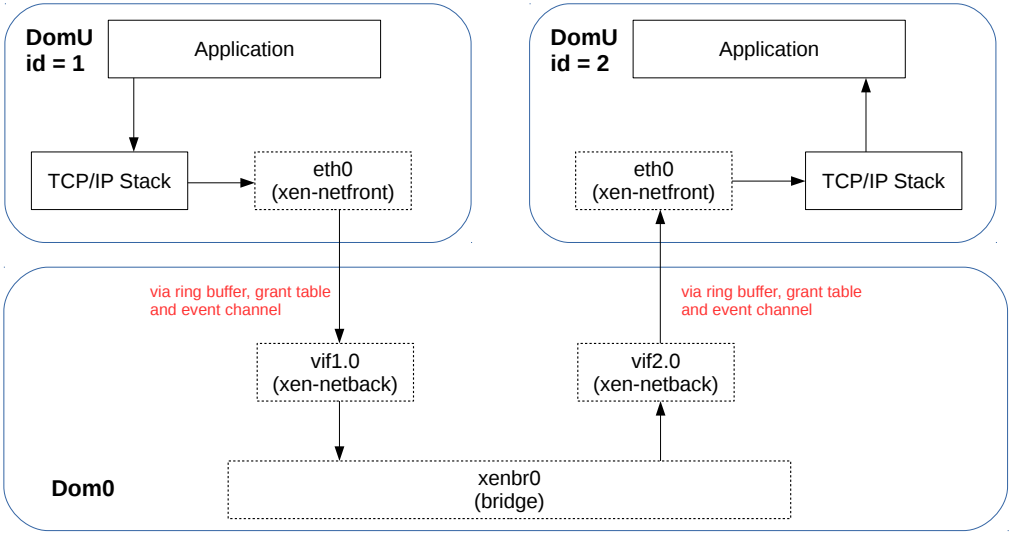
Unbreakable Enterprise Kernel v4.1.12-89

- drivers/net/xen-netfront.c
- drivers/net/xen-netback/xenbus.c
- drivers/net/xen-netback/netback.c
- drivers/net/xen-netback/interface.c

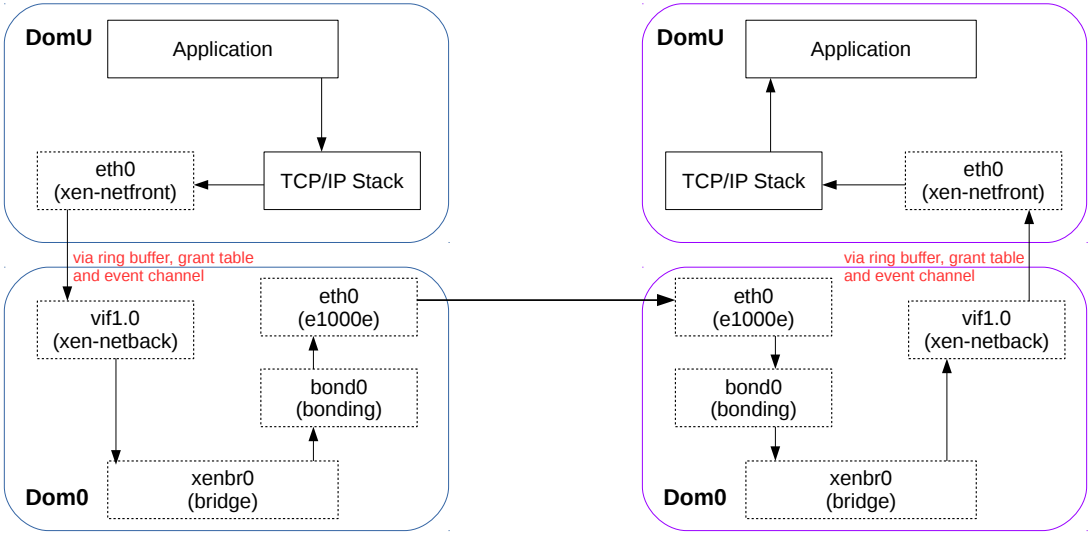
kernel upstream v4.9-rc8

- drivers/net/xen-netfront.c
- drivers/net/xen-netback/xenbus.c
- drivers/net/xen-netback/netback.c
- drivers/net/xen-netback/interface.c
- drivers/net/xen-netback/rx.c
- drivers/net/xen-netback/hash.c

Paravirtual networking scenario 1/2



Paravirtual networking scenario 2/2



PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	
device discovery	PCI Tree	
device configuration	PCI Config Space (IO/MMIO)	
data flow	DMA Ring Buffer	
shared memory	N/A or IOMMU	
interrupt	IOAPIC, MSI, MSI-X	

PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	
device configuration	PCI Config Space (IO/MMIO)	
data flow	DMA Ring Buffer	
shared memory	N/A or IOMMU	
interrupt	IOAPIC, MSI, MSI-X	

PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	Xenstore
device configuration	PCI Config Space (IO/MMIO)	
data flow	DMA Ring Buffer	
shared memory	N/A or IOMMU	
interrupt	IOAPIC, MSI, MSI-X	

PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	Xenstore
device configuration	PCI Config Space (IO/MMIO)	Xenstore
data flow	DMA Ring Buffer	
shared memory	N/A or IOMMU	
interrupt	IOAPIC, MSI, MSI-X	

PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	Xenstore
device configuration	PCI Config Space (IO/MMIO)	Xenstore
data flow	DMA Ring Buffer	Memory Ring Buffer
shared memory	N/A or IOMMU	
interrupt	IOAPIC, MSI, MSI-X	

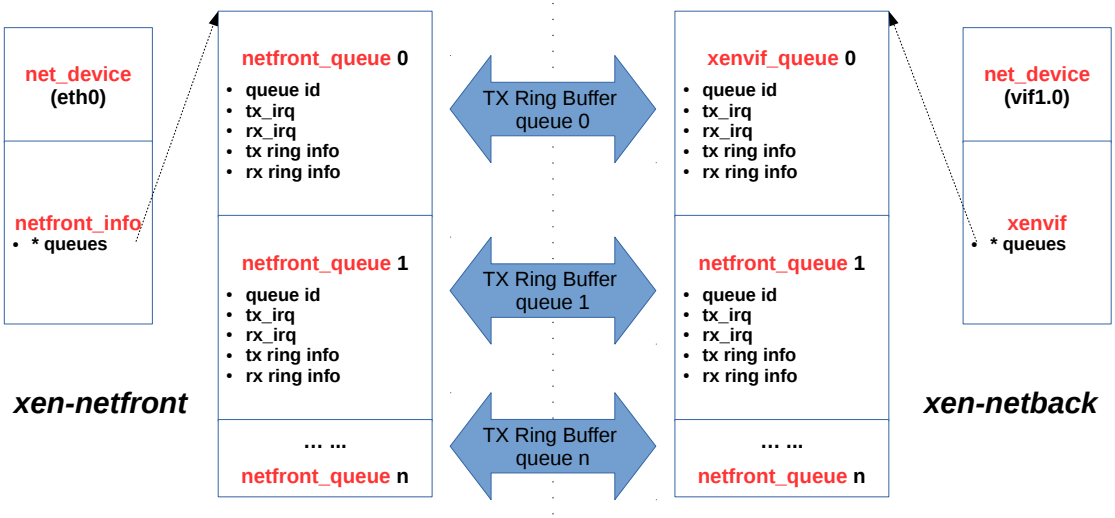
PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	Xenstore
device configuration	PCI Config Space (IO/MMIO)	Xenstore
data flow	DMA Ring Buffer	Memory Ring Buffer
shared memory	N/A or IOMMU	Grant Table
interrupt	IOAPIC, MSI, MSI-X	

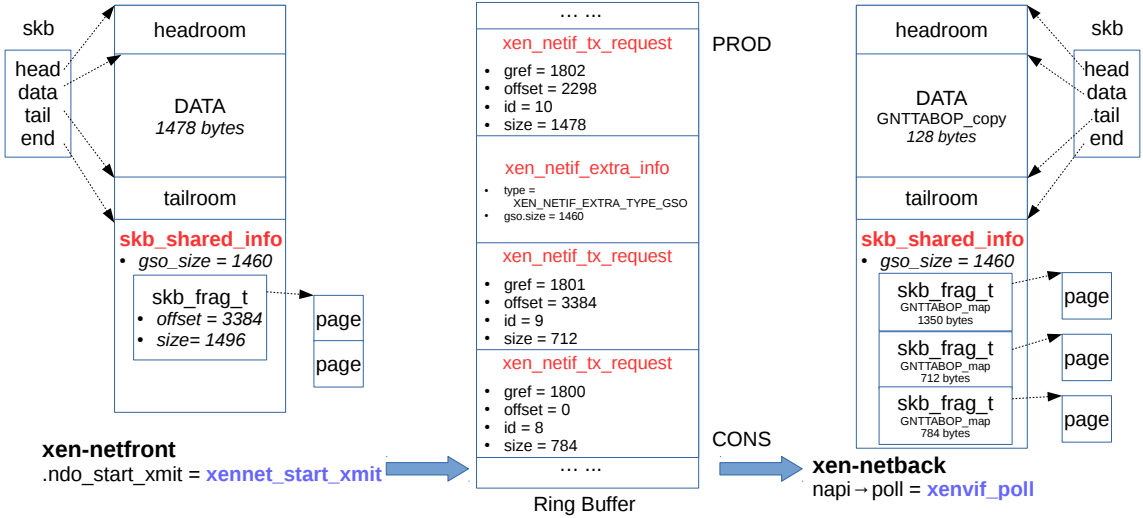
PV driver vs. PCI driver

	PCI driver	PV driver
device abstraction	pci_device, pci_driver	xenbus_device, xenbus_driver
device discovery	PCI Tree	Xenstore
device configuration	PCI Config Space (IO/MMIO)	Xenstore
data flow	DMA Ring Buffer	Memory Ring Buffer
shared memory	N/A or IOMMU	Grant Table
interrupt	IOAPIC, MSI, MSI-X	Event Channel

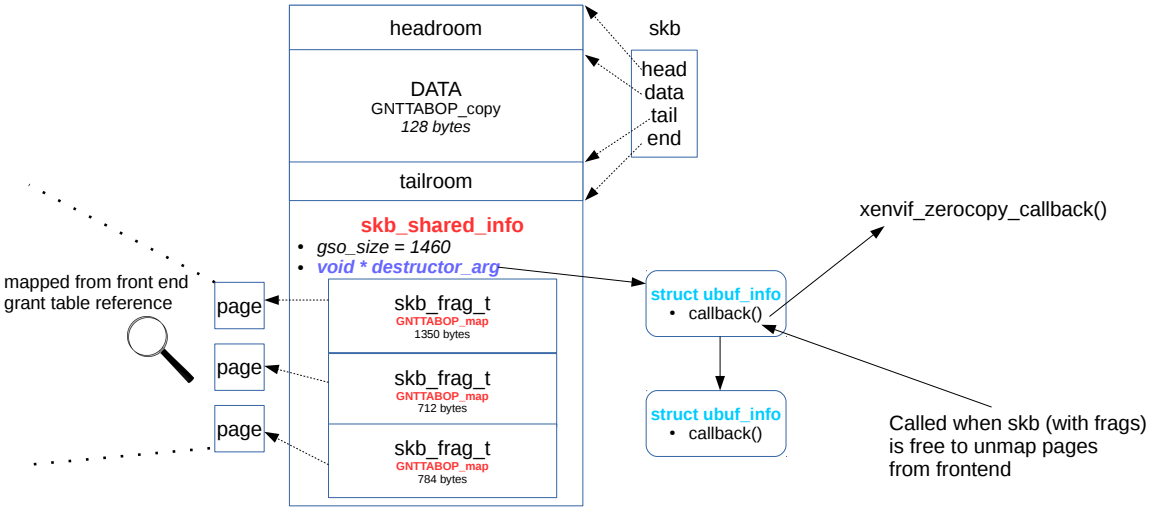
pv xmit: front —> backend 1/3



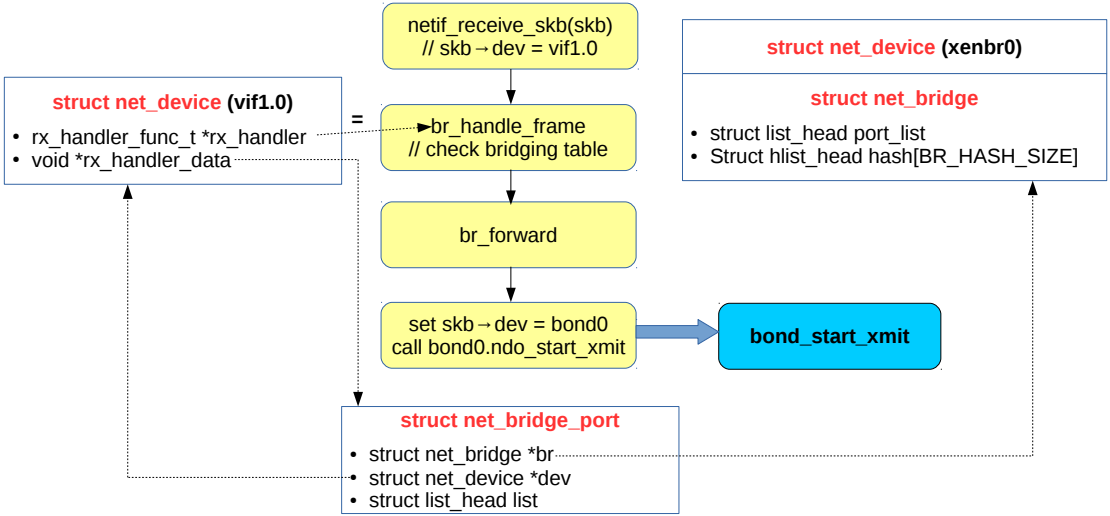
pv xmit: front —> backend 2/3



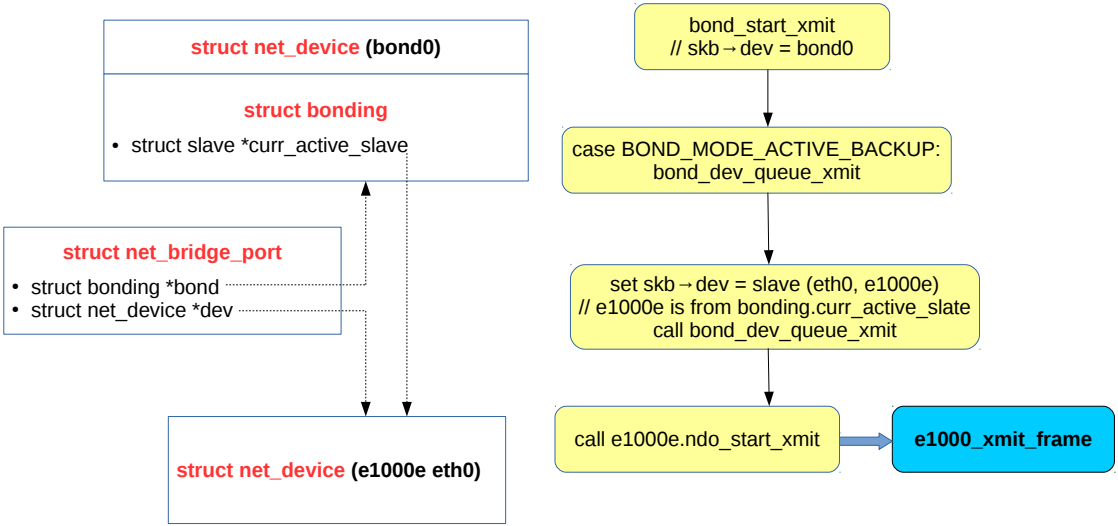
pv xmit: front —> backend 3/3



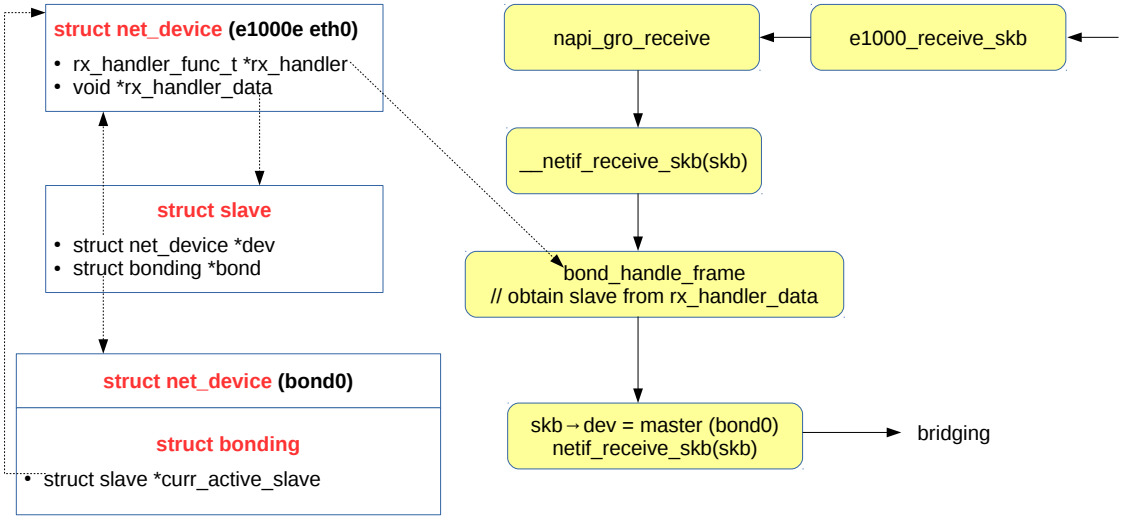
pv xmit: backend —> bridge



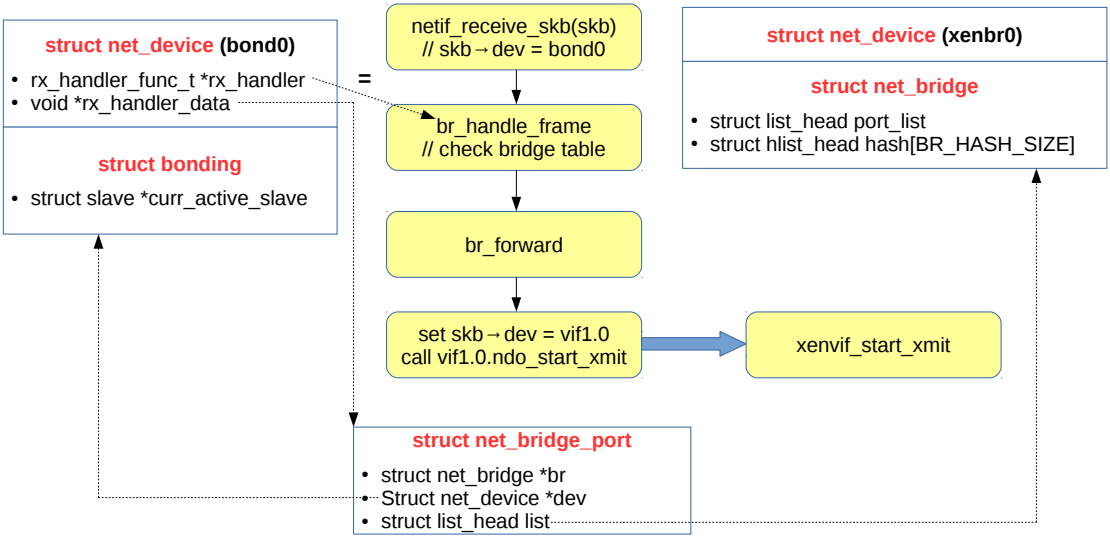
pv xmit: bridge —> bond —> physical NIC



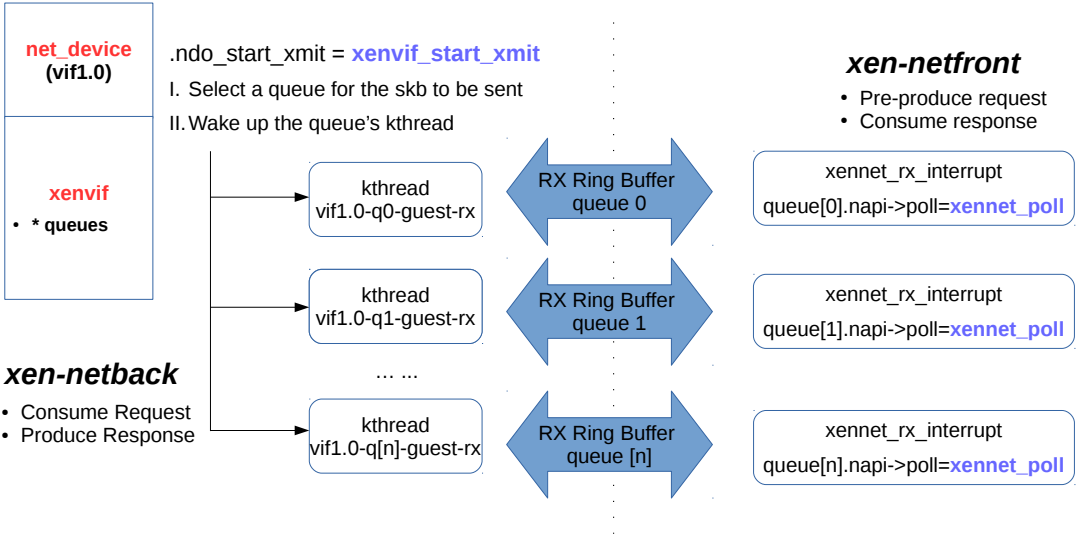
pv recv: physical NIC —> bond —> bridge



pv recv: bridge —> backend



pv recv: backend —> frontend



xen-netfront/xen-netback summary: req/rsp protocol

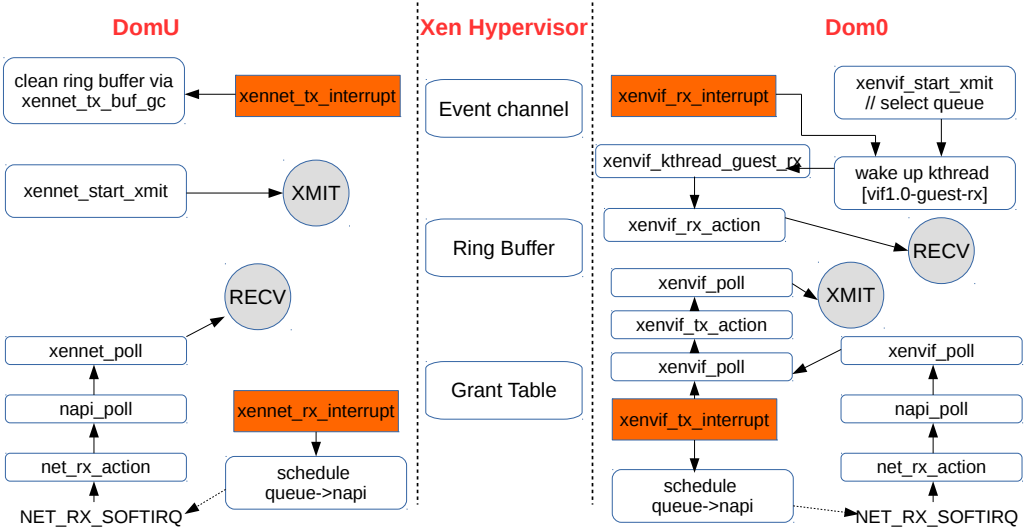
netfront to netback (produce req)

1. 1st page of linear data (skb->data)
2. extra info (xen_netif_extra_info)
3. the rest of linear data (skb->data)
4. all skb fragments (skb_shinfo(skb)->frags)

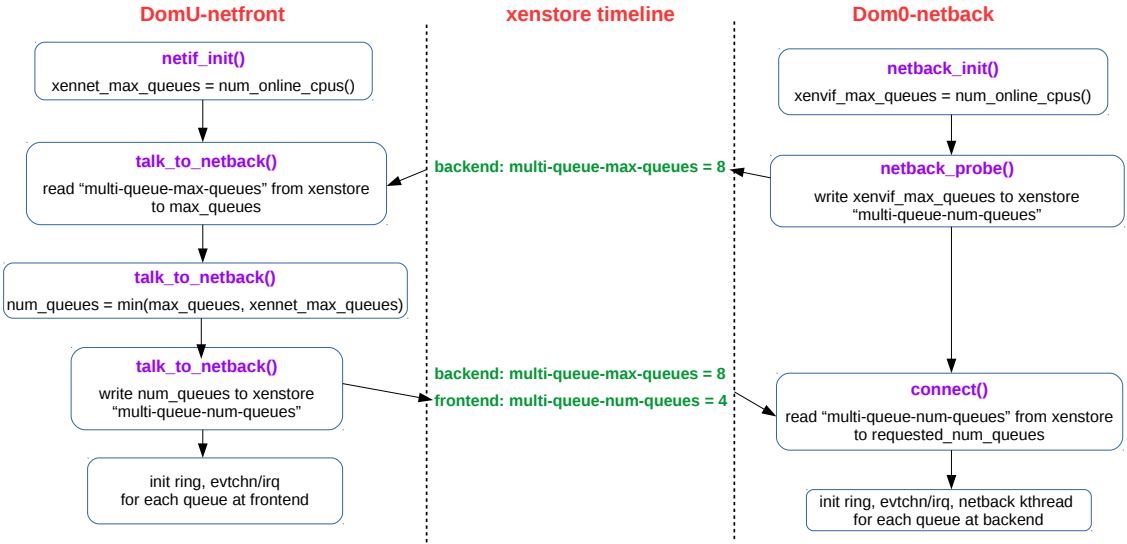
netback to netfront (produce rsq)

1. 1st page of linear data (skb->data)
2. extra info (xen_netif_extra_info)
3. the rest of linear data (skb->data)
4. all skb fragments (skb_shinfo(skb)->frags)

xen-netfront/xen-netback summary: irq and napi



features: multiqueue (default)



features: gso/tso offload

- Segmentation Offload
  - GSO (Generic Segmentation Offload): software segmentation
  - TSO (TCP Segmentation Offload): hardware segmentation

features: gso/tso offload

- Segmentation Offload
  - GSO (Generic Segmentation Offload): software segmentation
  - TSO (TCP Segmentation Offload): hardware segmentation
- TSO would postpone segmentation to as late (low level) as possible

features: gso/tso offload

- Segmentation Offload
  - GSO (Generic Segmentation Offload): software segmentation
  - TSO (TCP Segmentation Offload): hardware segmentation
- TSO would postpone segmentation to as late (low level) as possible
- TSO info is shared via "struct xen_netif_extra_info gso" in ring buffer
  - gso.gso->u.gso.size = skb_shinfo(skb)->gso_size;
  - gso->u.gso.type = XEN_NETIF_GSO_TYPE_TCPV6;

features: gso/tso offload

- Segmentation Offload
  - GSO (Generic Segmentation Offload): software segmentation
  - TSO (TCP Segmentation Offload): hardware segmentation
- TSO would postpone segmentation to as late (low level) as possible
- TSO info is shared via "struct xen_netif_extra_info gso" in ring buffer
  - gso.gso->u.gso.size = skb_shinfo(skb)->gso_size;
  - gso->u.gso.type = XEN_NETIF_GSO_TYPE_TCPV6;
- TSO and other offload features are stored in xenstore (e.g., feature-gso-tcpv4)
  - .ndo_fix_features = xennet_fix_features
  - .ndo_set_features = xennet_set_features

features: gso/tso offload

- Segmentation Offload
  - GSO (Generic Segmentation Offload): software segmentation
  - TSO (TCP Segmentation Offload): hardware segmentation
- TSO would postpone segmentation to as late (low level) as possible
- TSO info is shared via "struct xen_netif_extra_info gso" in ring buffer
  - gso.gso->u.gso.size = skb_shinfo(skb)->gso_size;
  - gso->u.gso.type = XEN_NETIF_GSO_TYPE_TCPV6;
- TSO and other offload features are stored in xenstore (e.g., feature-gso-tcpv4)
  - .ndo_fix_features = xennet_fix_features
  - .ndo_set_features = xennet_set_features
- checksum offload
  - XEN_NETTXF_csum_blank: Protocol checksum field is blank in the packet (hardware offload)
  - XEN_NETTXF_data_validated: Packet data has been validated against protocol checksum

features: multicast

DomU

Please support multicast

xenstore

request-multicast-control = 1

I will support multicast

Dom0

XEN_NETIF_EXTRA_TYPE_MCAST_ADD

XEN_NETIF_EXTRA_TYPE_MCAST_DEL

xen_netif_extra_info

struct list_head fe_mcast_addr

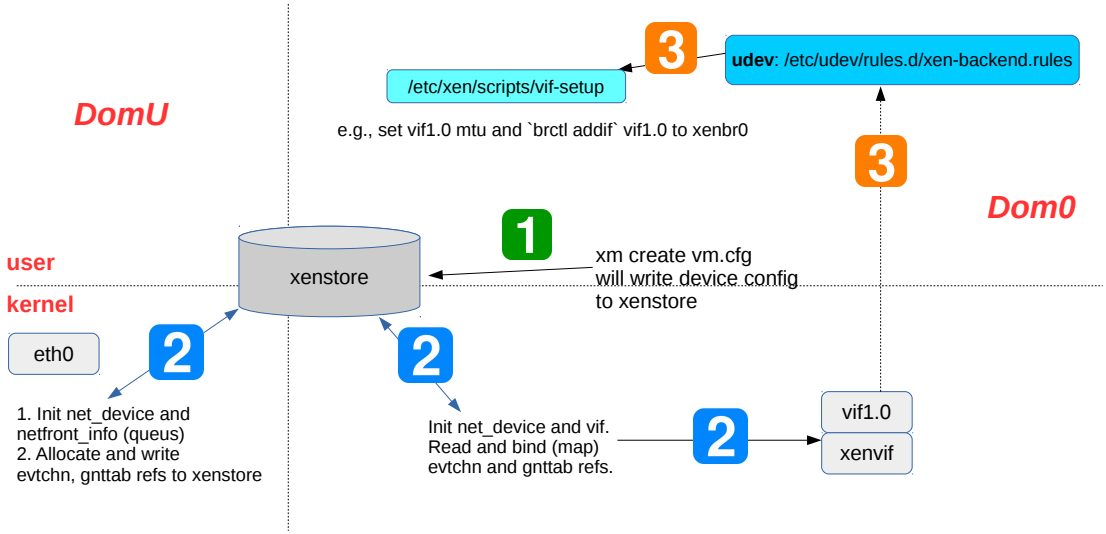
check registered multicast address
and drop if not registered

```
if (!xenvif_mcast_match(vif, eth->h_dest))
    goto drop;
```

xenvif_start_xmit()

SKB

xen-netfront/xen-netback init



performance tuning

- netfront/netback multiqueue
- Limit dom0 CPUs to first NUMA socket
- Interrupt affinity to reduce CPU 0 workload
- VCPU affinity to improve memory access performance
- Jumbo frame
- NIC offload
- TCP Parameter Settings

interesting works related to paravirtual I/O

- Achieving 10 Gb/s Using Safe and Transparent Network Interface Virtualization. VEE 2009
- Efficient and Scalable Paravirtual I/O System. USENIX ATC 2013
- A Comprehensive Implementation and Evaluation of Direct Interrupt Delivery. VEE 2015
- vRIO: Paravirtual remote I/O. ASPLOS 2016
- Hash, don't cache (the page table). SIGMETRICS 2016

Take-Home Message

- Xen paravirtual networking workflow

Take-Home Message

- Xen paravirtual networking workflow
- Xen paravirtual networking framework

Take-Home Message

- Xen paravirtual networking workflow
- Xen paravirtual networking framework
- Xen paravirtual networking init, protocol, features

Take-Home Message

- Xen paravirtual networking workflow
- Xen paravirtual networking framework
- Xen paravirtual networking init, protocol, features
- Xen paravirtual networking performance