

Linux on System z Networking

Stefan Raspl
IBM

2/5/2013
12374

(C) 2012, 2013 IBM Corporation



Trademarks



The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

AIX*	IBM*	PowerVM	System z10	z/OS*
BladeCenter*	IBM eServer	PR/SM	WebSphere*	zSeries*
DataPower*	IBM (logo)*	Smarter Planet	z9*	z/VM*
DB2*	InfiniBand*	System x*	z10 BC	z/VSE
FICON*	Parallel Sysplex*	System z*	z10 EC	
GDPS*	POWER*	System z9*	zEnterprise	
HiperSockets	POWER7*			

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license there from.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Windows Server and the Windows logo are trademarks of the Microsoft group of countries.

InfiniBand is a trademark and service mark of the InfiniBand Trade Association.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

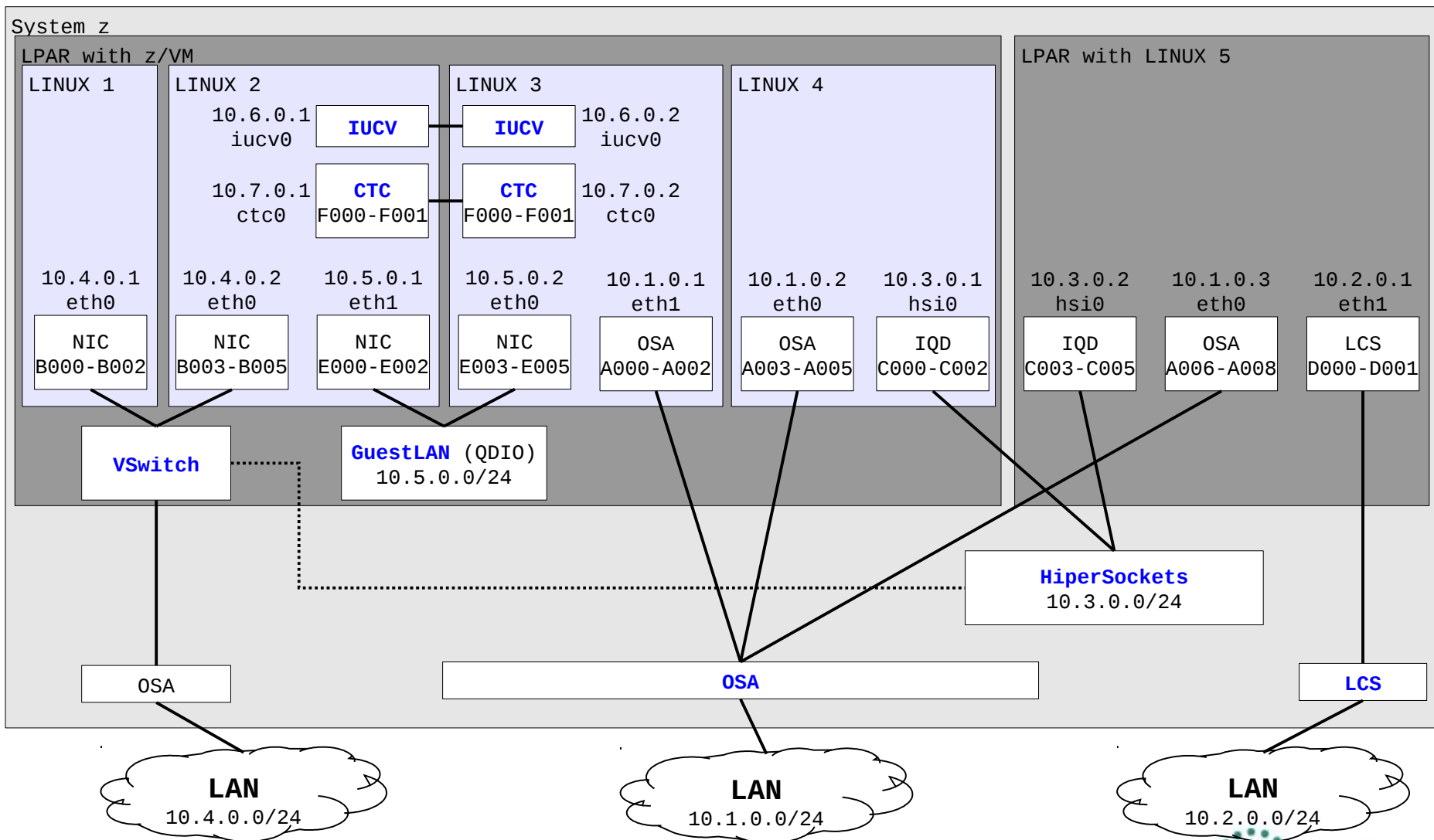
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.



Agenda

- Linux on System z Networking Devices and Drivers
- Configuration of Network Devices
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Network Example



QETH Device Driver

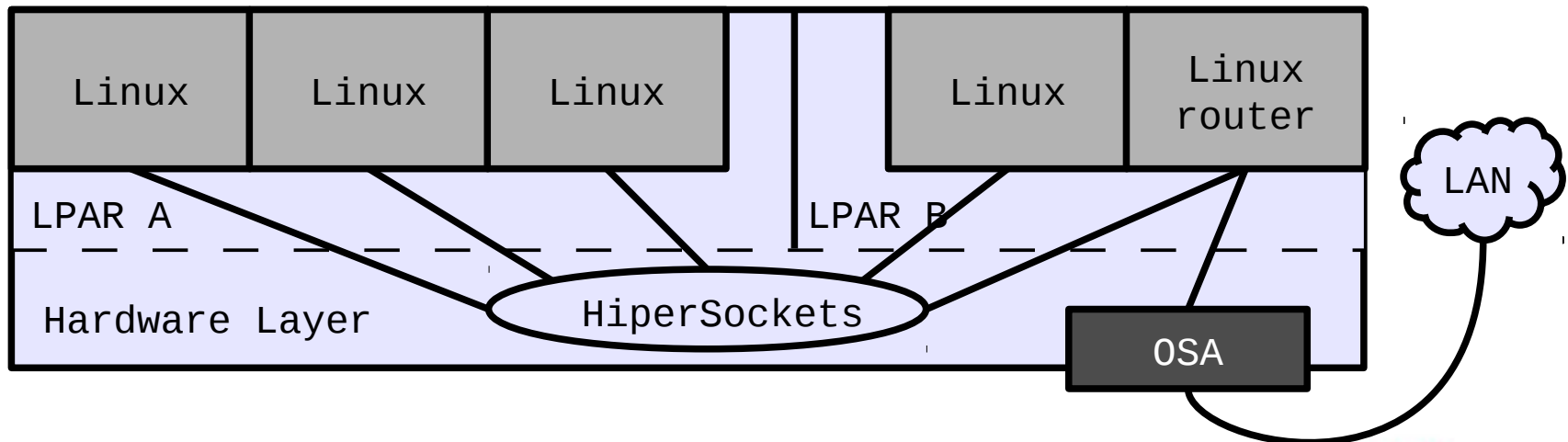
- Supports
 - OSA Express / OSA Express2 / OSA Express3 / OSA Express 4 – OSD type (=QDIO)
 - Fast/Giga/10GBit Ethernet (fiber infrastructure)
 - 1000Base-T Ethernet (copper infrastructure)
 - System z HiperSockets
 - z/VM
 - GuestLAN:
 - Type QDIO (layer2 / layer3)
 - Type Hipersockets
 - z/VM VSWITCH (layer2 / layer3)
 - IPv4, IPv6, VLAN, Channel Bonding
 - VIPA, Proxy ARP, IP Address Takeover: Special support in layer3 mode
- Primary network driver for Linux on System z
- Main focus in current and future development

HiperSockets and OSA-Express Features

Feature	EC12	z196, z114	System z10	System z9	zSeries
HiperSockets	Yes	Yes	Yes	Yes (layer3)	Yes (layer3)
OSA-Express4S	1 Gb Eth. 10 Gb Eth.	1 Gb Eth. 10 Gb Eth.	Not supp.	Not supported	Not supported
OSA-Express3	1 Gb Eth. 10 Gb Eth. 1000Base-T Eth.	1 Gb Eth. 10 Gb Eth. 1000Base-T Eth.	1 Gb Eth. 10 Gb Eth. 1000Base-T Eth.	Not supported	Not supported
OSA-Express2	Not supp.	1 Gb Eth. 1000Base-T Eth.	1 Gb Eth. 10 Gb Eth. 1000Base-T Eth.	1 Gb Eth. 10 Gb Eth. 1000Base-T Eth.	Not supported
OSA-Express	Not supp.	Not supp.	Not supp.	Fast Ethernet 1 Gb Eth. 1000Base-T Eth.	Fast Ethernet 1 Gb Eth. 1000Base-T Eth. Token Ring ATM

System z Hipersockets

- Connectivity within a central processor complex without physical cabling
- Licensed Internal Code (LIC) function
emulating Data Link Layer of an OSA-device (internal LAN)
- Internal Queued Input/Output (IQDIO) at memory speed
- 4 different MTU sizes supported:
 - 8KB, 16KB, 32KB, 56KB
- Support of
 - Broadcast, VLAN, IPv6, Layer2 (starting with z10)



Virtual Switch

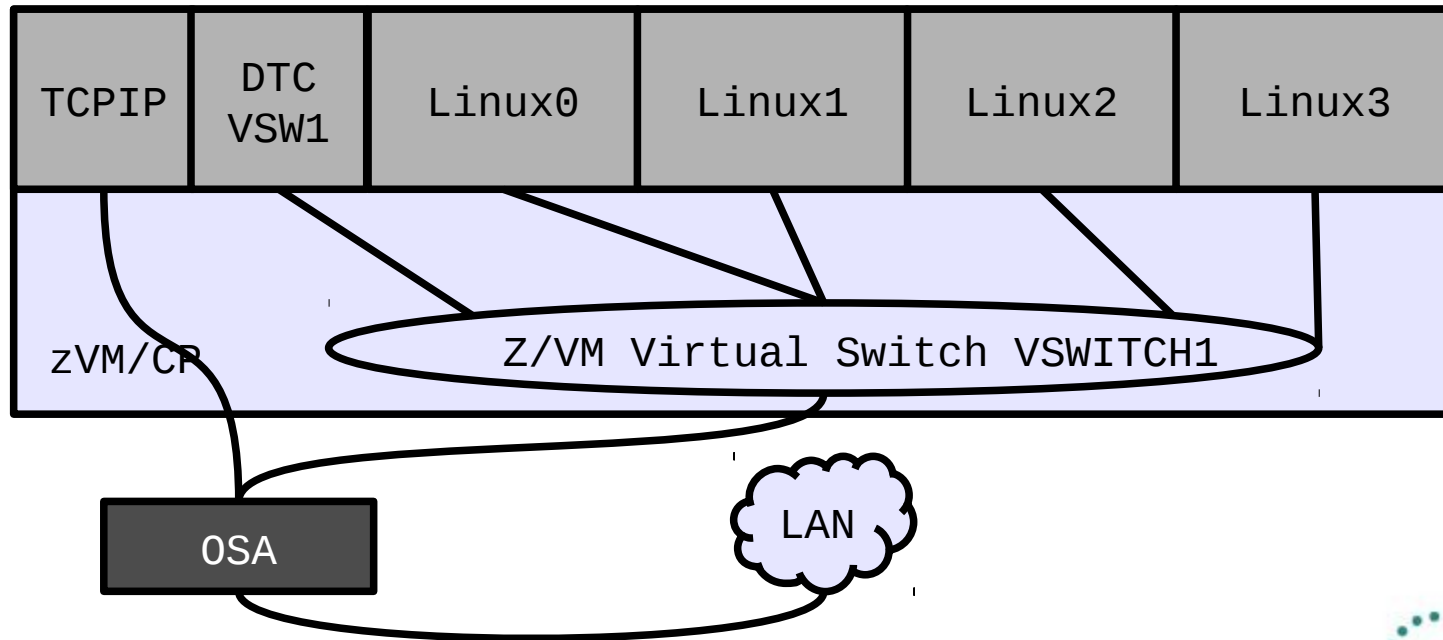
- Create simulated Layer 2 or Layer 3 switch device
- VM access control and VLAN authorization
- Create ports
- Connect NIC to Virtual Switch (LAN Segment)
- Full MAC address management (generation and assignment)
- 1-n VSWITCHs per z/VM Image

Create VSWITCH from PRIVCLASS B User ID

```
DEF VSWITCH VSWITCH1 ETHERNET
SET VSWITCH VSWITCH1 GRANT {user ID}
```

From Linux Virtual Machines

```
DEF NIC 600 TYPE QDIO
COUPLE 600 SYSTEM VSWITCH1
```



z/VM GuestLANs and VSWITCH

z/VM Guest LAN

- A simulated LAN segment
- Types:
 - QDIO:
 - **IP:** layer3, IPv4 only
 - **Ethernet:** layer2, IPv4 or IPv6
 - HiperSockets: IPv4/layer3 only
- No physical connection
- Unrestricted / restricted
- Persistent / transient
- As many as you want

z/VM VSWITCH

- Special purpose GuestLAN
- Type QDIO only
 - **IP:** layer3, IPv4 only
 - **Ethernet:** layer2, IPv4 or IPv6
- Built-in IEEE 802.1q bridge to outside network
- 1-8 associated OSA-connections
- Restricted
- Persistent
- Failover and Link Aggregation
- Port Isolation

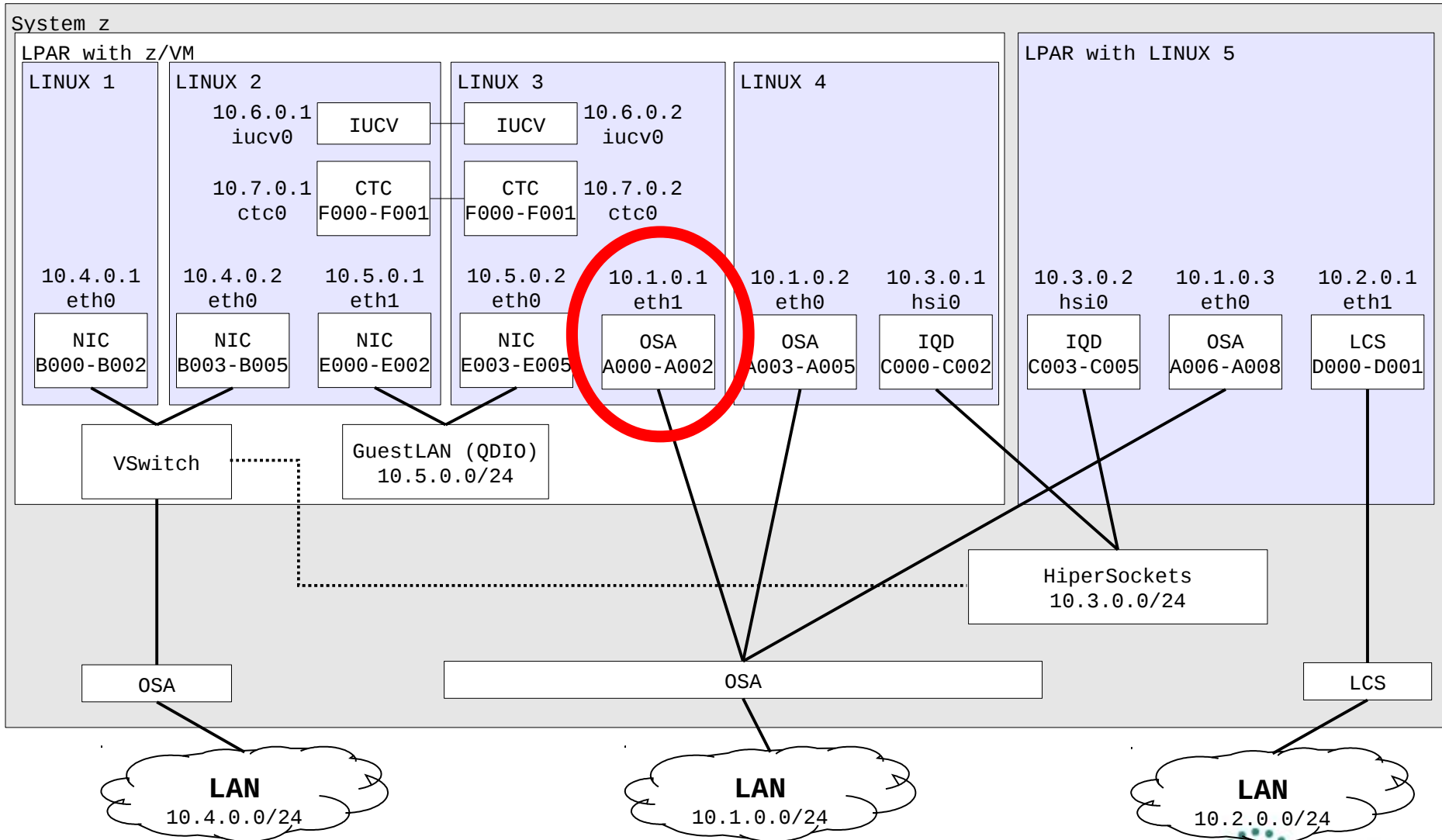
Virtual Network Devices – NICs (Virtual Network Interface Cards)

- Defined by directory or CP DEFINE NIC command
 - Type QDIO or HIPER (must match LAN type)
- The only thing visible to Linux

Agenda

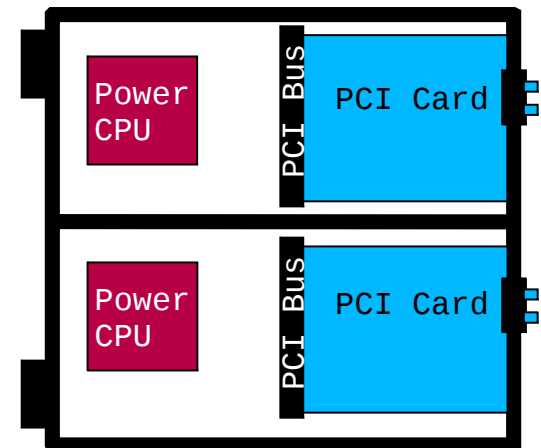
- Linux on System z Networking Devices and Drivers
- **Generic Configuration of Network Devices**
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Networking Device Configuration – Example



Primary Network Device: OSA Express

- 'Integrated Power computer' with network daughter card
- Shared between up to 640 OSA devices
- Three device numbers (ccw devices) per OSA device:
 - Read device (control data ← OSA)
 - Write device (control data → OSA)
 - Data device (network traffic)
- OSA Address Table: which OS image has which IP address
- Network traffic Linux ↔ OSA, either
 - IP (layer3 mode)
 - One MAC address for all stacks
 - OSA handles ARP
(Address Resolution Protocol)
 - Ethernet / data link layer level (layer2 mode)



Network Device Configuration – Generic

- Load the device driver module

```
root@larsson:~> modprobe qeth
```

- Create a new device by grouping its CCW devices:

```
root@larsson:~> echo 0.0.a000,0.0.a001,0.0.a002 > \  
/sys/bus/ccwgroup/drivers/qeth/group
```

- Set optional attributes:

```
root@larsson:~> echo 64 > /sys/devices/qeth/0.0.a000/buffer_count
```

- Set the device online:

```
root@larsson:~> echo 1 > /sys/devices/qeth/0.0.a000/online
```

– automatically assigns an interface name to the qeth device:

- eth[n] for OSA devices
- hsi[n] for HyperSocket devices

- Configure an IP address and bring the device up:

```
root@larsson:~> ip addr add 10.1.0.1/24 dev eth0  
root@larsson:~> ip link set dev eth0 up
```

- **Note:** Use distro specific tools for persistent configuration

Agenda

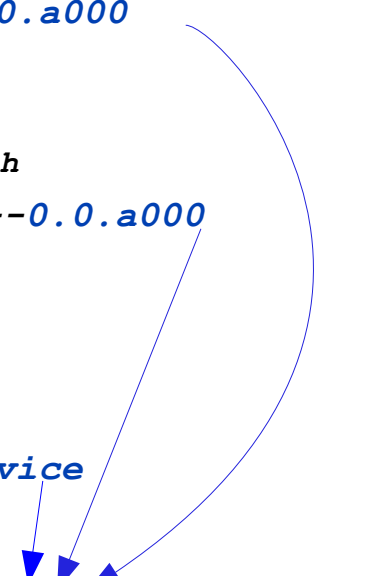
- Linux on System z Networking Devices and Drivers
- Generic Configuration of Network Devices
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Linux 2.6 Device Model – System z

```

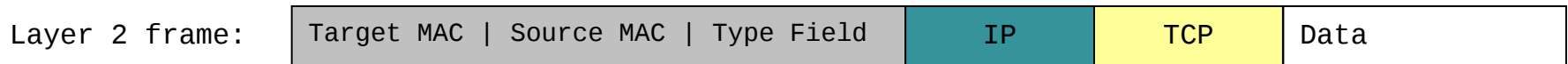
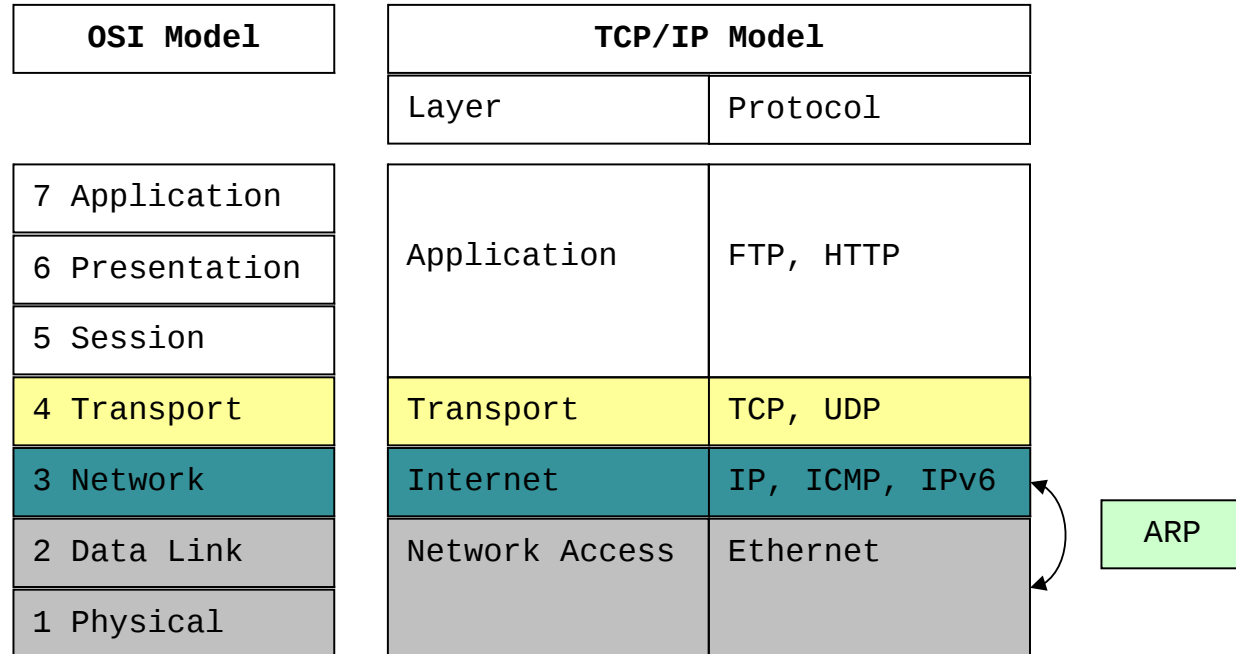
/sys
|--block
|  |--dasda
|  |...
|--bus
|  |--ccw
|  |--ccwgroup
|  |  |--devices
|  |    |--0.0.a000
|  |    |--drivers
|  |    |--lcs
|  |    |--qeth
|  |      |--0.0.a000
|  |--css
|--class
|  |--net
|    |--eth0
|      |--device
|--devices
|  |--qeth
|    |--0.0.a000

```



- Block subsystem (view):
Block devices and partitions
(dasda, sda, ram0)
- Bus subsystem (view):
Device drivers and devices sorted
by bus (ccw)
CCW Group Devices:
QETH, LCS
Example: a QETH device
- Class subsystem (view):
Logical devices sorted by type,
i.e. to which class they belong;
Logical devices have link to
hardware device
- Devices subsystem (view):
All the devices of a system

QETH Layer 3 vs Layer 2 mode



OSA-devices in layer3 mode: Offload Checksumming and TCP Segmentation

- TCP Checksumming: Additional redundancy check to protect data integrity
- Offload *checksumming* for **inbound** IP packages from Linux stack to OSA-card (activated per default)

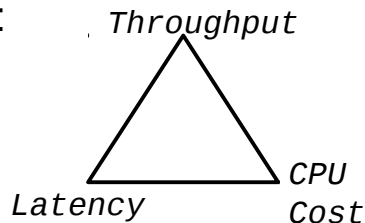
```
root@larsson:~> ethtool -K eth0 rx on
```

- Offload *checksumming* for **outbound** IP packages from Linux stack to OSA-card
 - Will not work for hosts reachable through *shared* usage of OSA express adapter
 - Saves *CPU cost*, but does not necessarily improve throughput (also see below)
- Offload **TCP segmentation** from Linux stack to OSA-card

```
root@larsson:~> ethtool -K eth0 tx on sg on tso on
```

- Move workload from Linux to OSA-Express adapter
 - Saves **CPU costs** if sending data blocks > MTU size
 - Will not work for hosts reachable through *shared* usage of OSA-Express adapter

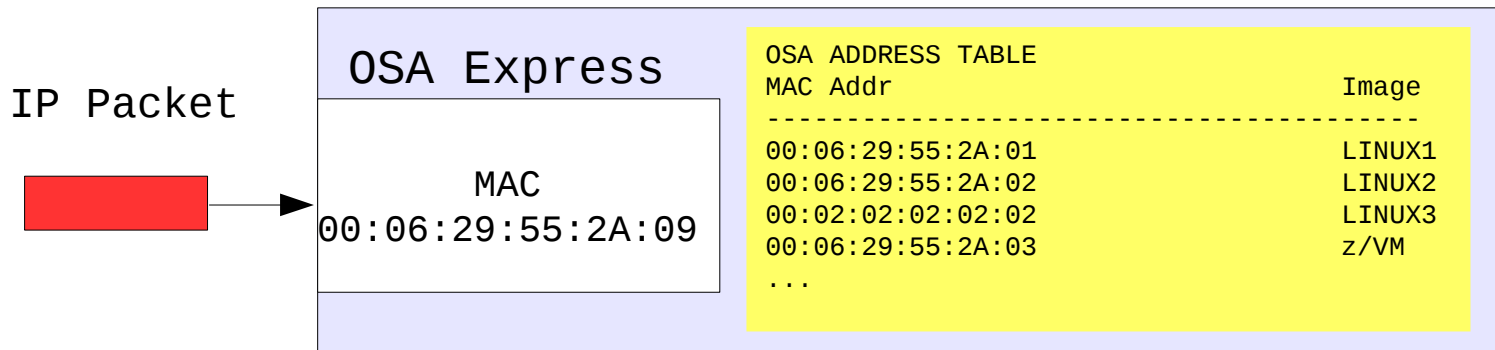
- But note:



- To investigate performance, use OSA/SF.

QETH Layer 2 mode

- OSA works with MAC address ==> no longer stripped from packets



- SLES10: hwcfg-qeth... file: QETH_LAYER2_SUPPORT=1
ifcfg-qeth... file: LLADDR='<MAC Address>'
- SLES11: 51.qeth....rule: ATTR{layer2}="1"
ifcfg-qeth... file: LLADDR='<MAC Address>'
- RHEL5/6: ifcfg-... file: MACADDR='<MAC Address>'
OPTIONS='layer2=1'

QETH Layer 2 mode (cont'd)

- Direct attached OSA:
 - MAC address must be defined manually with ifconfig
`ifconfig eth0 hw ether 00:06:29:55:2A:01`
 - Restrictions: Older OSA-generation ($\leq$ z990):
Layer2 and Layer3 traffic can be transmitted over the same OSA CHPID, but not between two images sharing the same CHPID !
- Hipersockets:
 - New layer2 support starting with z10 - MAC address automatically generated
 - Simultaneous Layer2 and Layer3 traffic possible, but no conversion
- VSWITCH or GuestLAN under z/VM: MAC address created by z/VM

```
define lan <lanname> ... type QDIO ETHERNET
    define nic <vdev> QDIO
    couple <vdev> <ownerid> <lanname>
define vswitch <vswname> ... ETHERNET ...
    define nic <vdev> QDIO
    couple <vdev> <ownerid> <lanname>
```

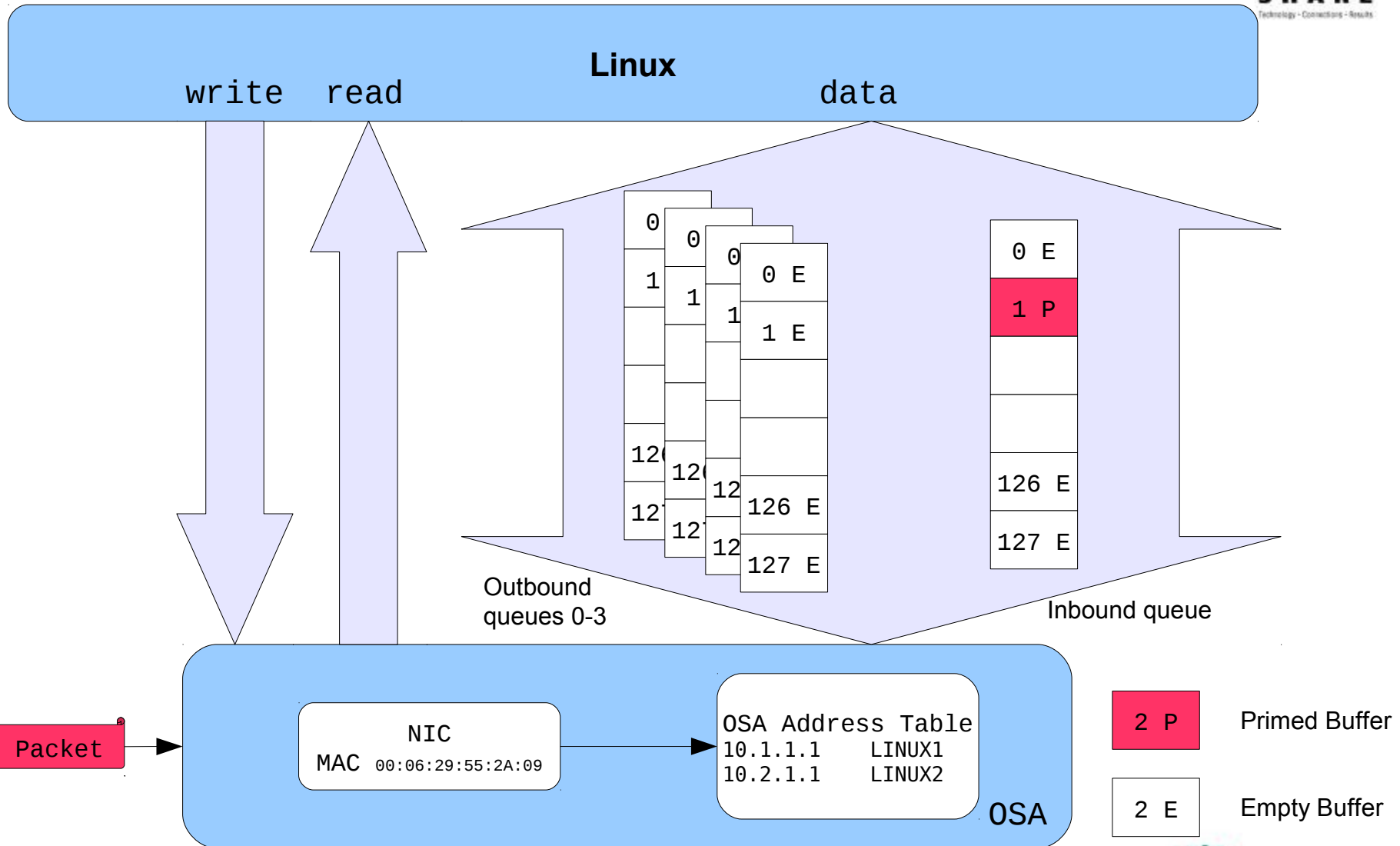
QETH Layer 2 mode (cont'd)

- Activating Layer 2 is done per device via sysfs attributes
- Possible layer2 values:
 - 0: use device in Layer 3 mode
 - 1: use device in Layer 2 mode

```
/sys  
|--devices  
   |--qeth  
      |--0.0.<devno>  
         |--layer2
```

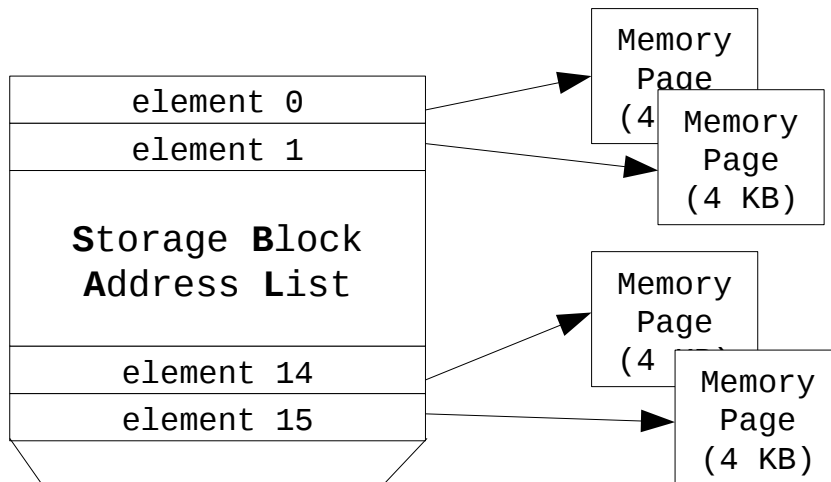
- Setting of layer2 attribute only permitted when device is offline!
- Advantages:
 - Independent of IP-protocol or any layer3 protocol
 - Channel bonding possible
 - tcpdump output readable without option fake_ll
 - No OSA-specific setup required for routing, proxy ARP

Queued Direct I/O (QDIO) Architecture



QETH Device sysfs Attribute `buffer_count`

- The number of allocated buffers for *inbound* QDIO traffic
→ **Memory usage.**



Per QETH device memory usage:

control data structures: ~ 200 KB
memory for one buffer: 64 KB

buffer_count = 8 --> ~ 712 KB

buffer_count = 128 --> ~ 8.4 MB

(Hipersocket numbers depend on MTU size)

0	1	2	3	...	124	125	126	127
---	---	---	---	-----	-----	-----	-----	-----

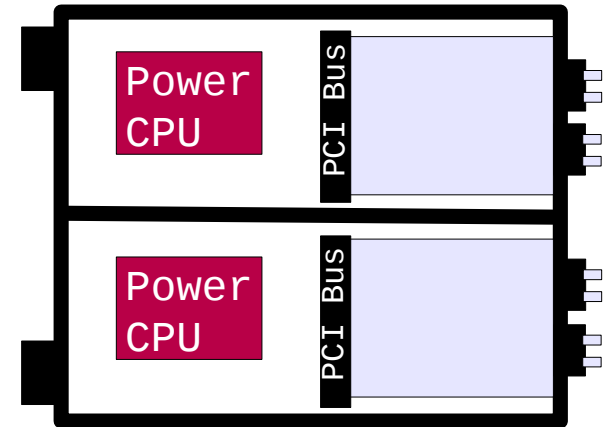
Save Memory
8 Buffers



Boost Performance
128 Buffers

OSA Express 3 – 2 ports within 1 chpid

- OSA Express2 – 2 CHPIDs with 1 port per CHPID
→ 2 ports in total
- OSA Express3 – 2 CHPIDs with 2 ports per CHPID
→ 4 ports in total ($\geq$ z10)
Models: OSA-Express3 GbE SX and LX



- New sysfs-attribute “portno” can contain '0' or '1'
- SLES10: hwcfg-qeth... file: QETH_OPTIONS="portno=1"
- SLES11: 51.qeth....rule: ATTR{portno}="1"
Or command: qeth_configure -n 1
- RHEL5/6: ifcfg-... file: OPTIONS='portno=1'

Agenda

- Linux on System z Networking Devices and Drivers
- Generic Configuration of Network Devices
- Further Networking Driver Aspects
- **System z related Networking Commands and Tools**
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Commands / tools for qeth-driven devices: lsqeth

- Brief list of known qeth devices: `lsqeth -p`

```
root@larsson:~> cat /proc/qeth
devices                                CHPID interface cardtype      port chksum
-----
0.0.a000/0.0.a001/0.0.a002 xA0   eth0      OSD_1000      0    SW
0.0.c000/0.0.c001/0.0.c002 xC0   hsi0      HiperSockets  0    SW
```

- Attributes of qeth device: `lsqeth` or `lsqeth <interface>`

```
root@larsson:~> lsqeth eth0
Device name                : eth0
-----
card_type                  : OSD_1000
cdev0                      : 0.0.a000
cdev1                      : 0.0.a001
cdev2                      : 0.0.a002
chpid                      : 76
online                     : 1
state                      : UP (LAN ONLINE)
buffer_count               : 16
layer2                     : 0
```

Commands / tools for qeth-driven devices: znetconf

- Allows the user to list, add, remove & configure System z network devices
- To list all configured network devices:

```
root@larsson:~> znetconf -c
```

Device IDs	Type	Card Type	CHPID	Drv.	Name	State
0.0.a000,0.0.a001,0.0.a002	1731/01	OSD_1000	76	qeth	eth0	online

- To list all potential network devices, use the **-u** option
- Configure device 0.0.a000 in layer2 mode and with portnumber “1”

```
root@larsson:~> znetconf -a a000 -o layer2=0 -o portno=1
```

- Remove network device 0.0.a000

```
root@larsson:~> znetconf -r a000
```

Commands / tools for geth-driven devices: ethtool

- Use ethtool to query, set and change attributes
- To query ethernet driver information:

```
root@larsson:~> ethtool -i eth0  
driver: geth_l3  
version: 1.0  
firmware-version: 0893  
bus-info: 0.0.a000/0.0.a001/0.0.a002
```

- To query the offload information of the specified ethernet device

```
root@larsson:~> ethtool -k eth0  
Offload parameters for eth0:  
rx-checksumming: off  
tx-checksumming: off .....
```

- Example: To change the inbound checksumming offload parameter

```
root@larsson:~> ethtool -K eth0 rx on
```

Agenda

- Linux on System z Networking Devices and Drivers
- Generic Configuration of Network Devices
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Outlook: New Utility qethcoat

- List address table, physical and logical device information
- Use to provide additional information in case of service calls instead of OSA/SF

```
root@larsson:~> qethcoat eth2
```

```
PCHID: 0x0240  
CHPID: 0x92  
Manufacturer MAC address: 00:14:5e:76:ed:26  
Configured MAC address: 00:00:00:00:00:00  
Data device sub-channel address: 0xe202  
CULA: 0x00  
Unit address: 0x02  
Physical port number: 0  
Number of output queues: 1  
Number of input queues: 1  
Number of active input queues: 0  
CHPID Type: OSD  
Interface flags: 0x00000000  
OSA Generation: OSA-Express3  
Port speed/mode: 1000 Mb/s / full duplex  
Port media type: copper  
Jumbo frames: yes  
Firmware: 0x00000085
```

```
IPv4 router: no  
IPv6 router: no  
IPv4 vmac router: no  
IPv6 vmac router: no  
Connection isolation: not active  
IPv4 assists enabled: 0x00000000  
IPv6 assists enabled: 0x00215c60  
IPv4 outbound checksum enabled: 0x00000000  
IPv6 outbound checksum enabled: 0x00000000  
IPv4 inbound checksum enabled: 0x00000000  
IPv6 inbound checksum enabled: 0x00000000  
L2 vmac: 02:00:00:f2:e3:bb
```

```
gmac  
----  
33:33:00:00:00:01  
01:00:5e:00:00:01  
33:33:ff:f2:e3:bb
```

Outlook: VEPA Mode

- **Virtual Edge Port Aggregator**
- Send all packets to adjacent switch for consistent enforcement of security policy
- Other motivation cited elsewhere is to offload packet switching from SW to HW
 - Pointless on System z, since shared OSA already provides HW switching
- Requires switch port to be configured in reflective relay (RR) mode
- Activate using 'isolation' sysfs attribute:

```
root@larsson:~> echo forward > /sys/devices/qeth/0.0.e200/isolation
```

- Note: Previously, port isolation mode 'forward' was effectively identical to 'drop'. Not true anymore!

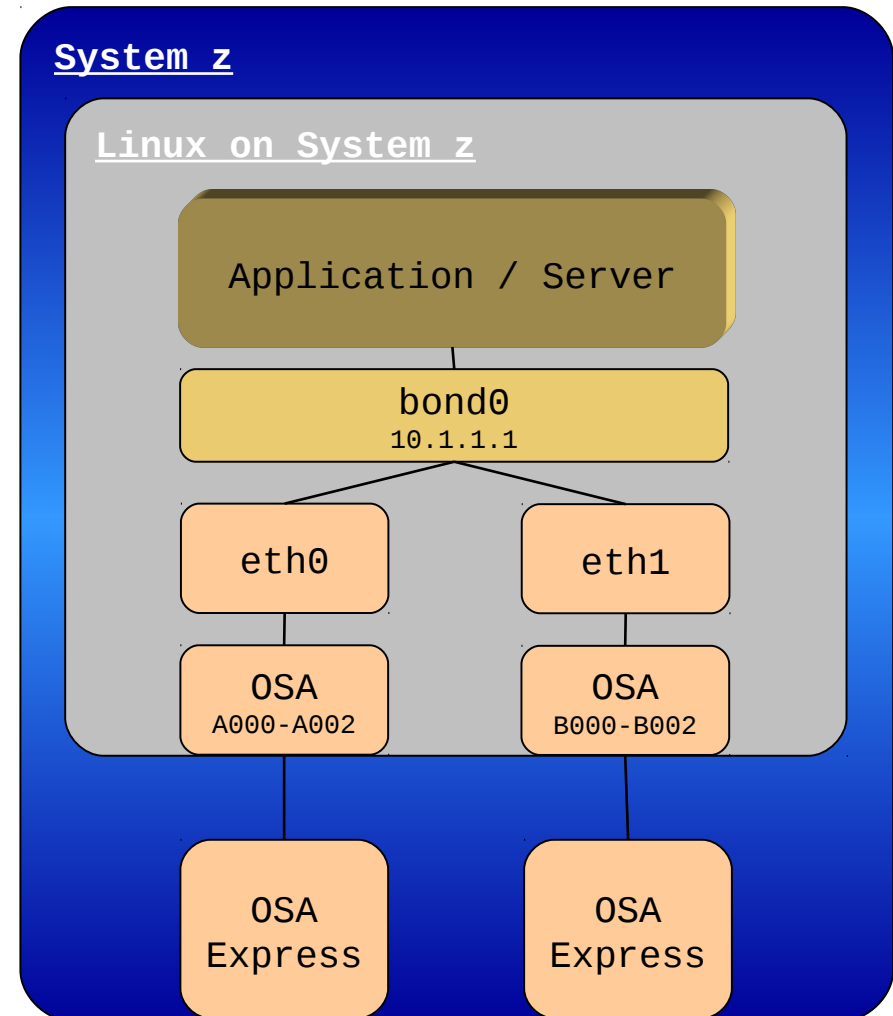
Agenda

- Linux on System z Networking Devices and Drivers
- Generic Configuration of Network Devices
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- **Channel Bonding**
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Channel Bonding Support

- The Linux bonding driver provides a method for aggregating multiple network interfaces into a single, logical “bonded” interface
- Provides failover and/or load-balancing functionality
- Better performance depending on bonding mode
- Requires layer2 devices
- Further information

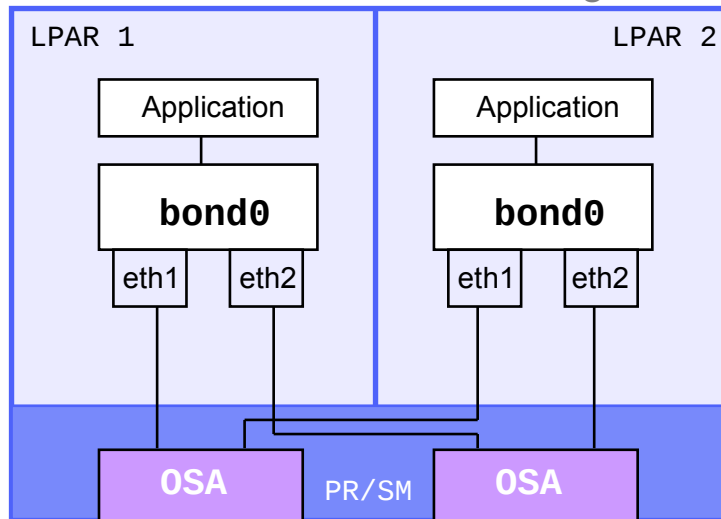
<http://sourceforge.net/projects/bonding>



Leveraging Virtualization for Network Interface Redundancy and Automated Failover

Resource Virtualization:

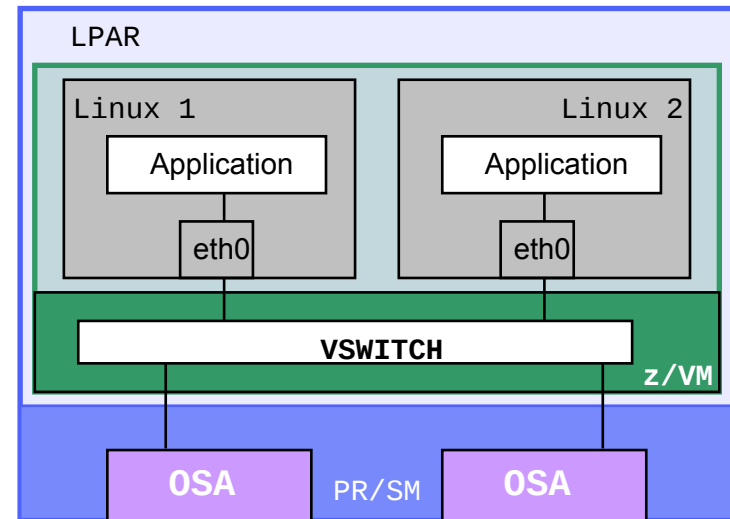
OSA Channel Bonding



- Linux *bonding* driver enslaves multiple OSA connections to create a single logical network interface card (NIC)
- Detects loss of NIC connectivity and automatically fails over to surviving NIC
- Active/backup & aggregation modes
- Separately configured for each Linux**

System Virtualization:

•z/VM VSWITCH



- z/VM *VSWITCH* enslaves multiple OSA connections. Creates virtual NICs for each Linux guest
- Detects loss of physical NIC connectivity and automatically fails over to surviving NIC
- Active/backup & aggregation modes
- Centralized configuration benefits all guests**

Setting Up Channel Bonding

- Load bonding module with miimon option (enables link monitoring)

```
root@larsson:~> modprobe bonding miimon=100 mode=balance-rr
```

- Add MAC addresses to slave devices eth0 & eth1 (not necessary for VSWITCH)

```
root@larsson:~> ifconfig eth0 hw ether 00:06:29:55:2A:01  
root@larsson:~> ifconfig eth1 hw ether 00:05:27:54:21:04
```

- Activate the bonding device bond0

```
root@larsson:~> ifconfig bond0 10.1.1.1 netmask 255.255.255.0
```

- Connect slave devices eth0 & eth1 to bonding device bond0

```
root@larsson:~> ifenslave bond0 eth0 eth1
```

Setting Up Channel Bonding (SLES10/11)

- Interface configuration file for a slave device
SLES10: */etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.a000*
SLES11: */etc/sysconfig/network/ifcfg-eth0*

```
BOOTPROTO='static'  
IPADDR=' '  
SLAVE='yes'  
STARTMODE='onboot'  
LLADDR='00:06:29:55:2A:01'
```

Setting Up Channel Bonding (SLES10/11) (cont'd)

- Interface configuration file for a master device
/etc/sysconfig/network/ifcfg-bond0

```
BOOTPROTO='static'  
BROADCAST='10.1.1.255'  
IPADDR='10.1.1.1'  
NETMASK='255.255.255.0'  
NETWORK='10.1.1.0'  
STARTMODE='onboot'  
BONDING_MASTER='yes'  
BONDING_MODULE_OPTS='mode=active_backup fail_over_mac=active  
miimon=100'  
LLADDR='00:06:29:55:2A:03'  
# SLES10  
BONDING_SLAVE0='qeth-bus-ccw-0.0.a000'  
BONDING_SLAVE1='qeth-bus-ccw-0.0.b000'  
# SLES11  
BONDING_SLAVE0='eth0'  
BONDING_SLAVE1='eth1'
```

Setting Up Channel Bonding (RHEL5/RHEL6)

- Interface configuration file for a slave device
/etc/sysconfig/network/ifcfg-eth0

```
DEVICE=eth0
USERCTL='yes'
BOOTPROTO='none'
SLAVE='yes'
MASTER='bond0'
ONBOOT='yes'
SUBCHANNELS=0.0.a000,0.0.a001,0.0.a002
TYPE=Ethernet
OPTIONS="layer2=1"
MACADDR=00:06:29:55:2A:01
```

- Add / verify alias in module configuration file /etc/modprobe.conf (RHEL5 only)

```
...
alias bond0 bonding
options bond0 miimon=100 mode=active_backup fail_over_mac=active
...
```

Setting Up Channel Bonding (RHEL5/RHEL6) (cont'd)

- Interface configuration file for a master device
/etc/sysconfig/network/ifcfg-bond0

```
DEVICE=bond0
BOOTPROTO='none'
BROADCAST='10.1.1.255'
IPADDR='10.1.1.1'
NETMASK='255.255.255.0'
NETWORK='10.1.1.0'
ONBOOT='yes'
USERCTL='yes'
NETTYPE='qeth'
TYPE=Bonding
MACADDR=
# RHEL6 only
BONDING_OPTS='mode=active_backup fail_over_mac=active miimon=100'
```

Setting Up Channel Bonding (cont'd)

- Display the interface and bonding configuration

```
root@larsson:~> ifconfig
bond0      Link encap:Ethernet  HWaddr 00:06:29:55:2A:01
            inet addr:10.1.1.1  Bcast:10.255.255.255  ...

eth0       Link encap:Ethernet  HWaddr 00:06:29:55:2A:01
            UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500...

eth1       Link encap:Ethernet  HWaddr 00:06:29:55:2A:02
            UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500 ...
```

```
root@larsson:~> cat /proc/net/bonding/bond0
Bonding Mode: fault-tolerance (active-backup) (fail_over_mac active)
Primary Slave: None
Currently Active Slave: eth0
MII Status: up
MII Polling Interval (ms): 100
Slave Interface: eth0
MII Status: up
Permanent HW addr: 00:06:29:55:2A:01
Slave Interface: eth1
MII Status: up
Permanent HW addr: 00:06:29:55:2A:02
```

Agenda

- Linux on System z Networking Devices and Drivers
- Configuration of Network Devices
 - Generic (manual)
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem

Network Performance Results Summary

- Which connectivity to use:
 - External connectivity:
 - LPAR: 10 GbE cards
 - z/VM: VSWITCH with 10GbE card(s) attached
 - z/VM: For maximum throughput and minimal CPU utilization attach OSA directly to Linux guest
 - Internal connectivity:
 - LPAR: HiperSockets for LPAR-LPAR communication
 - z/VM: VSWITCH for guest-guest communication
- For highly utilized network devices consider
 - to use z/VM VSWITCH with link aggregation
 - to use channel bonding
 - that channel bonding for high availability has low overhead

Agenda

- Linux on System z Networking Devices and Drivers
- Generic Configuration of Network Devices
- Further Networking Driver Aspects
- System z related Networking Commands and Tools
- Outlook
- Channel Bonding
- Performance Aspects
- Frequent Problem
- Appendix: Distribution-specific Network Device Configuration
 - Novell/SUSE SLES10 and SLES11
 - RedHat RHEL5 and RHEL6

Frequent Problem: HiperSockets connection too slow

▪ ifconfig output

```
ifconfig hsi0
hsi0      Link encap:Ethernet  HWaddr 06:00:F2:01:00:1B
          inet addr:10.10.32.5  Bcast:10.10.63.255  Mask:255.255.224.0
          inet6 addr: fe80::400:f2ff:fe01:1b/64  Scope:Link
          UP BROADCAST RUNNING NOARP MULTICAST  MTU:32768  Metric:1
          RX packets:32285 errors:0 dropped:0 overruns:0 frame:0
          TX packets:44539 errors:382 dropped:0 overruns:0 carrier:0
```

▪ s390 debug feature

– Check for qeth errors:

```
cat /sys/kernel/debug/s390dbf/qeth_qerr
00 01316676190:387699 2 - 00 000003c001070a08 71 6f 75 74 65 72 72 00 | qouterr.
00 01316676190:387699 2 - 00 000003c00106ec82 20 46 31 35 3d 31 30 00 | F15=10.
00 01316676190:387699 2 - 00 000003c00106ec82 20 46 31 34 3d 30 30 00 | F14=00.
00 01316676190:387700 2 - 00 000003c00106ec82 20 71 65 72 72 3d 31 00 | qerr=1.
```

▪ dbginfo file

– Check for buffer count of receiving partner:

```
lsqeth hsi0 | grep buffer_count
buffer_count      : 16
```

Frequent Problem: HiperSockets connection too slow (cont'd)

- netstat -s

```
netstat -s
...
Tcp:
...
39409 segments received
51294 segments send out
70 segments retransmitted
0 bad segments received.
...
```

- Problem Origin: Input buffers of receiving partner are full:

- Too few input buffers
 - increase buffer_count to 128
- CPUs busy with something else
- LPAR share / VM guest share too low
- Increase sysctl values
 - net.ipv4.tcp_rmem, sample: 4096 131072 4194240
 - net.core.netdev_max_backlog, sample: 3000
- ...

- Imbalance between sender and receiver

References

- Linux on System z on developerWorks

<http://www.ibm.com/developerworks/linux/linux390>

- Linux on System z documentation

http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html

- Linux on System z – Downloads

http://www.ibm.com/developerworks/linux/linux390/development_recommended.html

- Linux on System z - Tuning Hints & Tips

<http://www.ibm.com/developerworks/linux/linux390/perf/index.html>

- IBM System z Connectivity Handbook

<http://www.redbooks.ibm.com/redpieces/abstracts/sg245444.html>

Questions?



Stefan Raspl

*Linux on System z
Development*

*Schönaicher Strasse 220
71032 Böblingen, Germany*

*Phone +49 (0)7031-16-2177
stefan.raspl@de.ibm.com*

Appendix: Distribution-specific Network Device Configuration

Network Device Configuration – SLES 10

Hardware **devices** ↔ Logical **interfaces**

Configuration files:

`/etc/sysconfig/hardware/`

`/etc/sysconfig/network/`

1:1 relationship

--> A hardware device always gets the right IP address

Naming convention:

`(hw|if)cfg-<device type>-bus-<bus type>-<bus location>`

e.g. `hwcfg-qeth-bus-ccw-0.0.a000`

`ifcfg-qeth-bus-ccw-0.0.a000`

Scripts:

`hwup / hwdown, ifup / ifdown`

see `/etc/sysconfig/hardware/skel/hwcfg-<device type>`

Network Device Configuration – SLES 10 (cont'd)

1. Create an OSA hardware device configuration file:
/etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.a000

```
CCW_CHAN_IDS='0.0.a000 0.0.a001 0.0.a002'  
CCW_CHAN_MODE='OSAPORT'  
CCW_CHAN_NUM='3'  
MODULE='qeth'  
MODULE_OPTIONS=''  
MODULE_UNLOAD='yes'  
SCRIPTDOWN='hwdown-ccw'  
SCRIPTUP='hwup-ccw'  
SCRIPTUP_ccw='hwup-ccw'  
SCRIPTUP_ccwgroup='hwup-qeth'  
STARTMODE='auto'  
QETH_LAYER2_SUPPORT='0'  
QETH_OPTIONS='checksumming=hw_checksumming'
```

Network Device Configuration – SLES 10 (cont'd)

2. Create an interface configuration file

/etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.a000

```
BOOTPROTO='static'  
BROADCAST='10.1.0.255'  
IPADDR='10.1.0.1'  
NETMASK='255.255.255.0'  
NETWORK='10.1.0.0'  
STARTMODE='onboot'
```

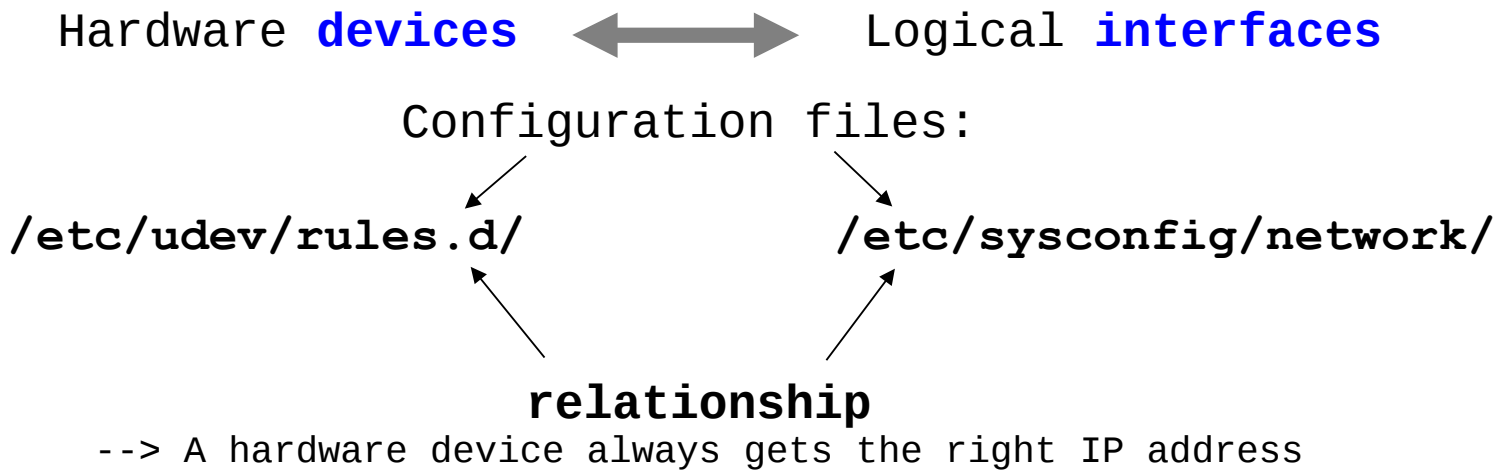
For more details about configuration variables see:

/etc/sysconfig/network/ifcfg.template

3. Before you reboot: Test the configuration, bring the device up and then the interface

```
root@larsson:~> hwup qeth-bus-ccw-0.0.a000  
root@larsson:~> ifup qeth-bus-ccw-0.0.a000
```

Network Device Configuration – SLES 11



Devices are configured via udev (framework for dynamic device configuration)

Naming convention:

51-**<device type>**-**<bus location>**.rules

70-persistent-net.rules

ifcfg-**<interface name>**

e.g. 51-**qeth**-bus-0.0.a000.rules

ifcfg-**eth0**

Scripts: `qeth_configure` and `ifup/ifdown`

Network Device Configuration – SLES 11 (cont'd)

1. “qeth_configure -p OSAPORT 0.0.a000 0.0.a001 0.0.a002 1” →
/etc/udev/rules.d/51-qeth-bus-0.0.a000.rules

```
# Configure qeth device at 0.0.a000/0.0.a001/0.0.a002
ACTION=="add", SUBSYSTEM=="drivers", KERNEL=="qeth",
IMPORT{program}="collect 0.0.a000 %k 0.0.a000 0.0.a001 0.0.a002 qeth"
ACTION=="add", SUBSYSTEM=="ccw", KERNEL=="0.0.a000",
IMPORT{program}="collect 0.0.a000 %k 0.0.a000 0.0.a001 0.0.a002 qeth"
ACTION=="add", SUBSYSTEM=="ccw", KERNEL=="0.0.a001",
IMPORT{program}="collect 0.0.a000 %k 0.0.a000 0.0.a001 0.0.a002 qeth"
ACTION=="add", SUBSYSTEM=="ccw", KERNEL=="0.0.a002",
IMPORT{program}="collect 0.0.a000 %k 0.0.a000 0.0.a001 0.0.a002 qeth"
TEST=="[ccwgroup/0.0.a000]", GOTO="qeth-0.0.a000-end"
ACTION=="add", SUBSYSTEM=="ccw", ENV{COLLECT_0.0.a000}=="0",
ATTR{[drivers/ccwgroup:qeth]group}="0.0.a000,0.0.a001,0.0.a002"
ACTION=="add", SUBSYSTEM=="drivers", KERNEL=="qeth", ENV{COLLECT_0.0.a000}=="0",
ATTR{[drivers/ccwgroup:qeth]group}="0.0.a000,0.0.a001,0.0.a002"
LABEL="qeth-0.0.a000-end"
ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.a000", ATTR{layer2}="0"
ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.a000",
ATTR{portname}="OSAPORT"
ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.a000", ATTR{portno}="0"
ACTION=="add", SUBSYSTEM=="ccwgroup", KERNEL=="0.0.a000", ATTR{online}="1"
```

Has to be first

Network Device Configuration – SLES 11 (cont'd)

2. Mapping between hardware device and Linux device
/etc/udev/rules.d/70-persistent-net.rules

```
...  
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="qeth", \  
    KERNELS=="0.0.a000", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"  
...
```

3. Create an interface configuration file
/etc/sysconfig/network/ifcfg-eth0 (For more details about configuration variables see: */etc/sysconfig/network/ifcfg.template*)

```
BOOTPROTO='static'  
BROADCAST='10.1.0.255'  
IPADDR='10.1.0.1'  
NETMASK='255.255.255.0'  
NETWORK='10.1.0.0'  
STARTMODE='onboot'
```

4. Before you reboot: Test the configuration

```
root@larsson:~> udevadm trigger  
root@larsson:~> ifup eth0
```

Network Device Configuration – RHEL 5

- Configuration files (RHEL 5)
/etc/modprobe.conf

```
alias eth0 qeth  
alias eth1 qeth  
alias hsi0 qeth  
alias eth2 lcs
```

*/etc/sysconfig/network-scripts/ifcfg-**<ifname>***

```
NETTYPE      qeth | lcs  
TYPE         Ethernet  
SUBCHANNELS  0.0.a000,0.0.a001,0.0.a002  
PORTNAME  
OPTIONS e.g. "layer2=1 portno=1"  
MACADDR
```

- **ifup/ifdown** scripts contain mainframe-specifics

Network Device Configuration – RHEL 5 (cont'd)

1. Create a network device configuration file:
/etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE=eth0  
SUBCHANNELS='0.0.a000 0.0.a001 0.0.a002'  
PORTNAME='OSAPORT'  
NETTYPE='qeth'  
TYPE='Ethernet'  
BOOTPROTO=static  
ONBOOT=yes  
BROADCAST='10.1.0.255'  
IPADDR='10.1.0.1'  
NETMASK='255.255.255.0'  
NETWORK='10.1.0.0'  
MACADDR='00:09:6B:1A:9A:89'  
OPTIONS='layer2=0'
```

Network Device Configuration – RHEL 5 (cont'd)

2. Add / verify alias in module configuration file
/etc/modprobe.conf

```
...  
alias eth0 qeth  
...
```

3. Before you reboot: Test the configuration

```
root@larsson:~> ifup eth0
```

For further information, see

<http://www.redhat.com/docs/manuals/enterprise>

Network Device Configuration – RHEL 6



1. Create a network device configuration file:
/etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE=eth0  
SUBCHANNELS='0.0.a000,0.0.a001,0.0.a002'  
PORTNAME='OSAPORT'  
NETTYPE='qeth'  
TYPE='Ethernet'  
BOOTPROTO=static  
ONBOOT=yes  
IPADDR='10.1.0.1'  
NETMASK='255.255.255.0'  
NETWORK='10.1.0.0'  
OPTIONS='layer2=0 portno=0'  
IPV6INIT=yes  
IPV6ADDR=fd00:10:30::30:31/80
```

Network Device Configuration – RHEL 6 (cont'd)

2. Mapping between hardware device and Linux device

/etc/udev/rules.d/70-persistent-net.rules

```
...  
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",  
ENV{INTERFACE_NAME}=="eth0", DRIVERS=="qeth",  
KERNELS=="0.0.a000", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"  
...
```

3. Before you reboot: Test the configuration

```
root@larsson:~> ifup eth0
```