IBM Linux and Technology Center

Networking with Linux on System z - Basic OSA Device Configuration

Martin Schwidefsky
IBM Lab Böblingen, Germany
March 2, 2011 – Session 8444

Trademarks & Disclaimer

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml: AS/400, DB2, e-business logo, ESCON, eServer, FICON, IBM, IBM Logo, iSeries, MVS, OS/390, pSeries, RS/6000, S/390, System Storage, System z9, VM/ESA, VSE/ESA, WebSphere, xSeries, z/OS, zSeries, z/VM.

The following are trademarks or registered trademarks of other companies

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries. LINUX is a registered trademark of Linux Torvalds in the United States and other countries. UNIX is a registered trademark of The Open Group in the United States and other countries. Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation. SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC. Intel is a registered trademark of Intel Corporation. * 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. References in this document to IBM products or services do not imply that IBM intends to make them available in every country. Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use. The information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice. Any references in this information to non-BM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

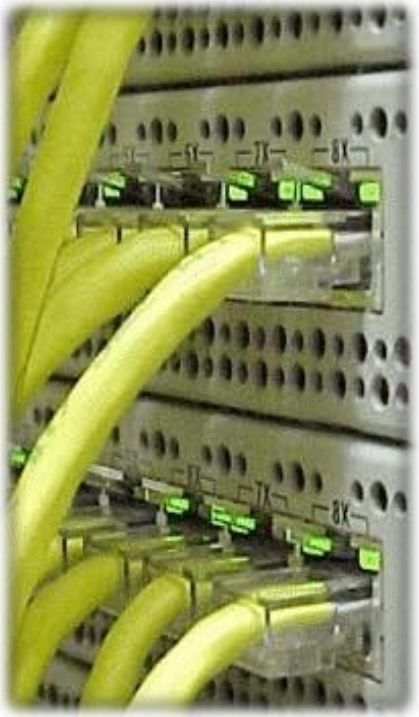
Agenda

- Linux on System z network device drivers
- Configuration of network devices
  - SUSE SLES10 and SLES11
  - RedHat RHEL5
  - Generic (manual)
- Further networking driver aspects
- Advanced aspects
  - Channel Bonding
  - Virtual IP Addresses
  - VLAN

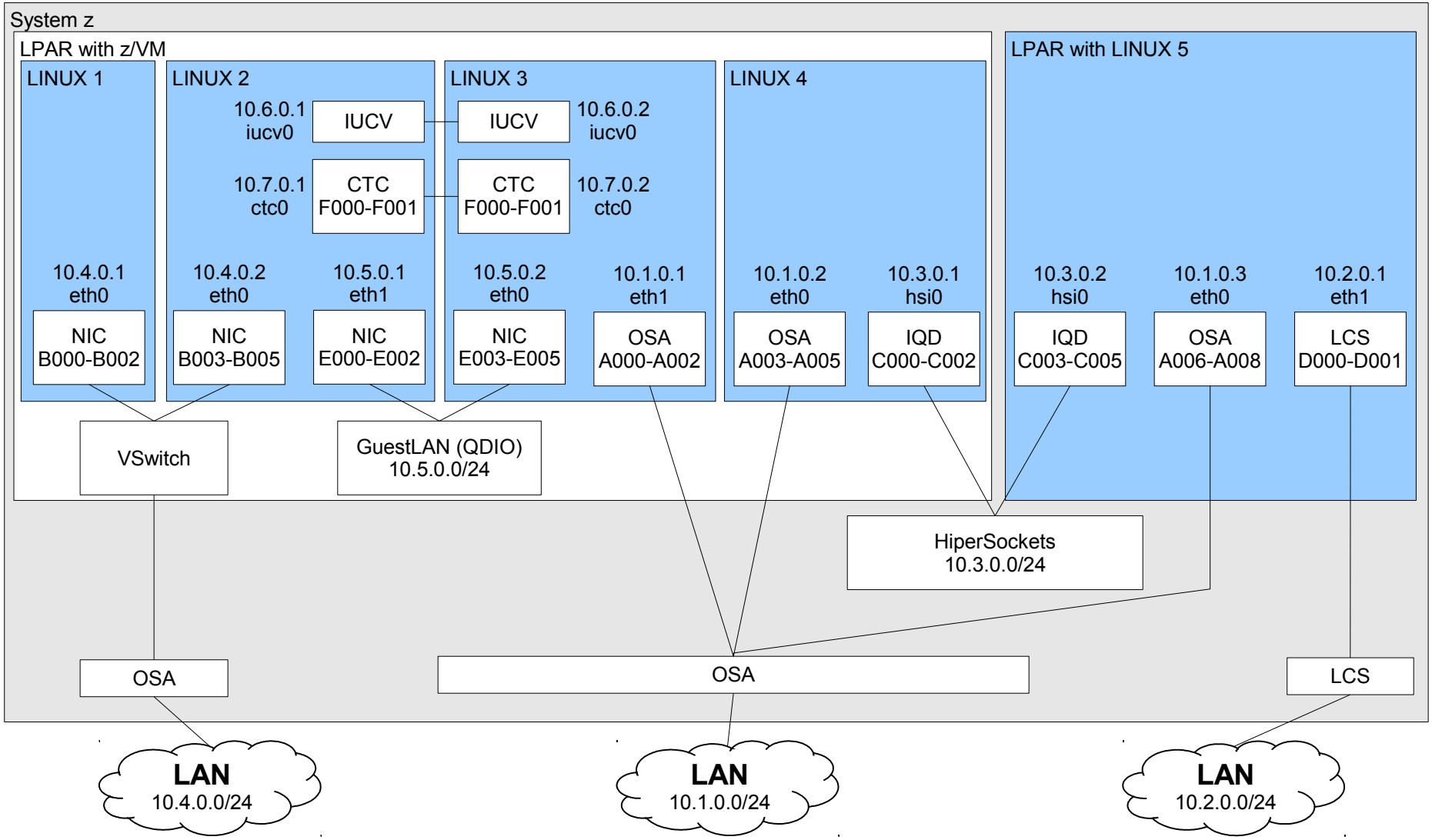
Network Device Drivers

Linux for System z Network Device Drivers

- **QETH**
- **LCS**
- **CTC(M) (stabilized)**
- **NETIUCV (stabilized)**
- **CLAW (stabilized)**



Network Example

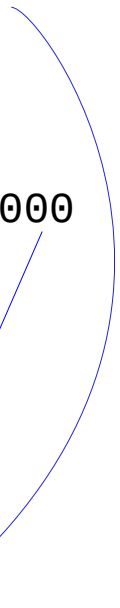


Linux 2.6 Device Model

- Integrated uniform device model that reflects a system's hardware structure
- Simplified device reference counting and locking
- Unified user interface via sysfs
  - Hierarchical, tree-like representation of system's hardware
  - Several subsystems provide different views of the hardware
  - Configuration of devices via attribute files
  - Dynamic attach/detach of devices possible

Linux 2.6 Device Model – System z Examples

```
/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 Devices:
DASD, RAM-Disk, Minidisk
SCSI, Loopback

CCW Group Devices:
QETH, LCS

Example: a QETH device

Many ways to find a device

LAN Channel Station (LCS) Device Driver

- Supports
  - OSA Express(2) (in non-QDIO mode OSE)
    - Fast Ethernet
    - 1000 Base-T Ethernet
    - HighSpeed TokenRing (<= z990)
    - ATM (running Ethernet LAN Emulation) (<= z990)
- May be preferred instead of QETH for security reasons
  - Administrator defines OSA Address Table, whereas with QETH each Linux registers its own IP address → restricted access

    But: performance is inferior to QETH's performance

Message to CTC and IUCV users

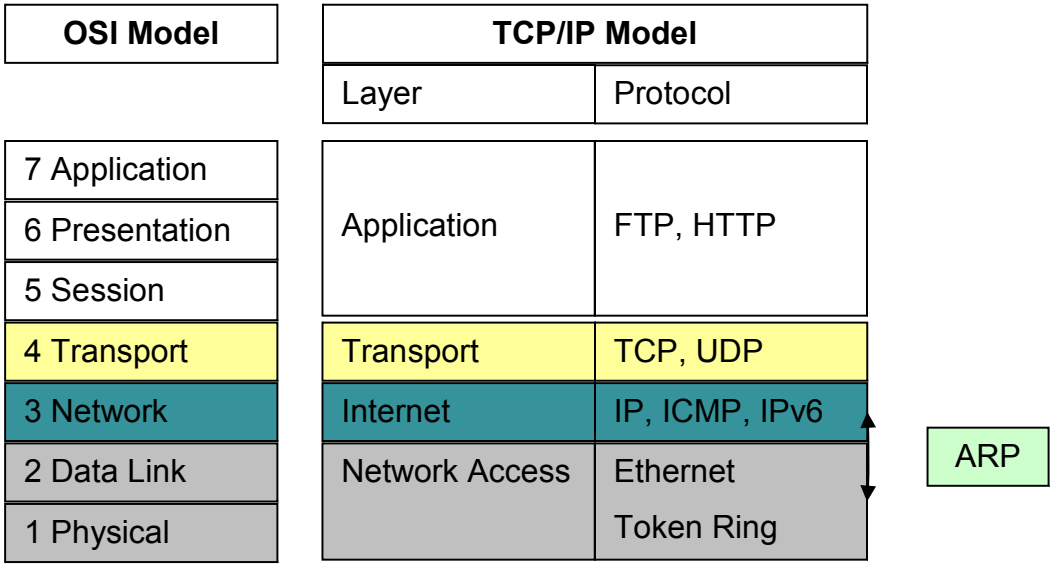
- CTC = Channel-to-Channel connection
- IUCV = Inter User Communication Vehicle
- CTC(M) and NETIUCV device drivers are deprecated (Linux 2.6+)
- Device drivers are still available for backward compatibility
- Please consider migration
  - Virtual CTC and IUCV (under z/VM) ==> guest LAN HiperSocket or guest LAN type QDIO
  - CTC inside a CEC ==> Hipersockets
  - CTC ==> OSA-Express (QDIO)

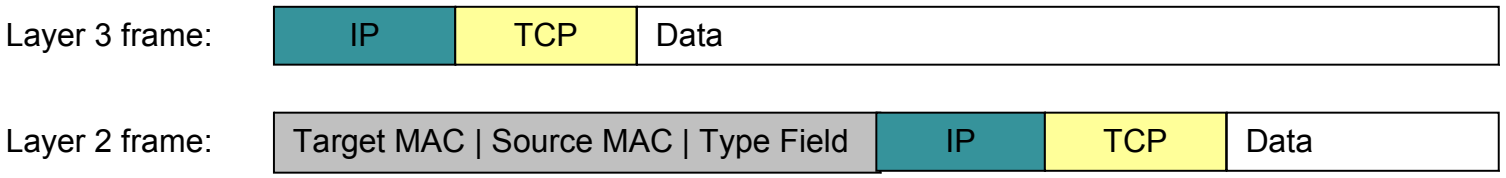
QETH Device Driver

- Supports
  - OSA Express / OSA Express2 / OSA Express3 – OSD type (=QDIO)
    - Fast/Giga/10GBit Ethernet (fiber infrastructure)
    - 1000Base-T Ethernet (copper infrastructure)
    - HighSpeed TokenRing (<= z990)
    - ATM (running Ethernet LAN Emulation) (<= z990)
  - System z HiperSockets
  - z/VM
    - GuestLAN Type QDIO (layer2 / layer3), Type Hiper
    - z/VM VSWITCH (layer2 / layer3)
  - IPv4, IPv6, VLAN, VIPA, Proxy ARP, IP Address Takeover, Channel Bonding

Primary network driver for Linux on System z
Main focus in current and future development

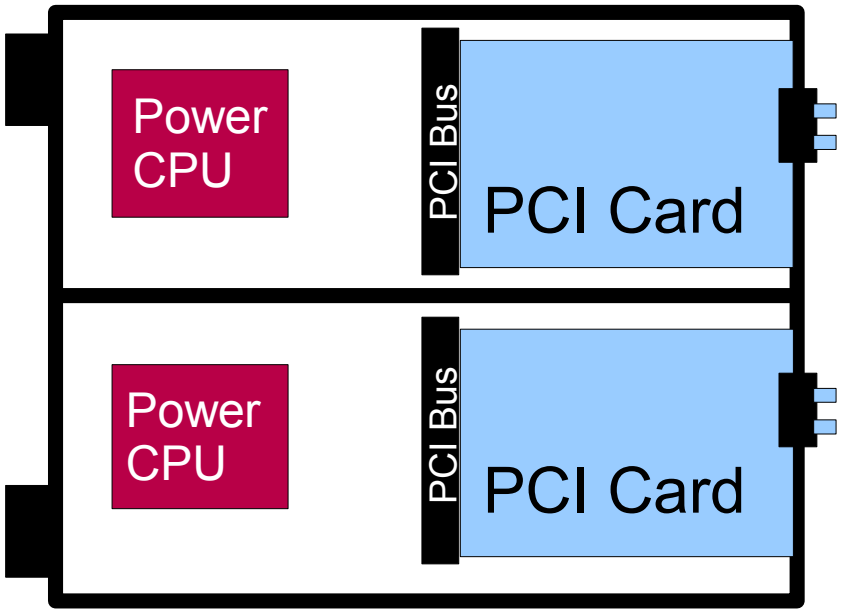
Layer 3 vs. Layer 2





Primary Network Device: OSA Express

- 'Integrated Power computer' with network daughter card
- Shared between up to 640 / 1920 TCP/IP stacks
- OSA Address Table: which OS image has which IP address
- Three devices (I/O subchannels) per stack:
  - Read device (control data ← OSA)
  - Write device (control data → OSA)
  - Data device (network traffic)
- Network traffic Linux ↔ OSA at IP (layer3) or Ethernet (layer2) level
- Layer 3:
  - One MAC address for all stacks
  - OSA handles ARP (Address Resolution Protocol)



System z HiperSockets

- Connectivity within a central processor complex without physical cabling
- Internal Queued Input/Output (IQDIO) at memory speed
- Licensed Internal Code (LIC) function
  emulating DataLink Layer of an OSA-device (internal LAN)
- 4 different maximum frame sizes / MTU sizes:

frame size	MTU size
16 KB	8 KB
24 KB	16 KB
40 KB	32 KB
64 KB	56 KB

- Support of
  - Broadcast
  - VLAN
  - IPv6
  - Layer2 (with z10)

z/VM GuestLANs and VSWITCH

- z/VM Guest LAN

- A simulated LAN segment
- Types:
  - QDIO: IPv4 and IPv6 (layer3)
  - Ethernet: lots of protocols (layer2)
  - HiperSockets: IPv4 and IPv6 (layer3)
- No physical connection
- Unrestricted / restricted
- Persistent / transient
- As many as you want

- **z/VM VSWITCH**

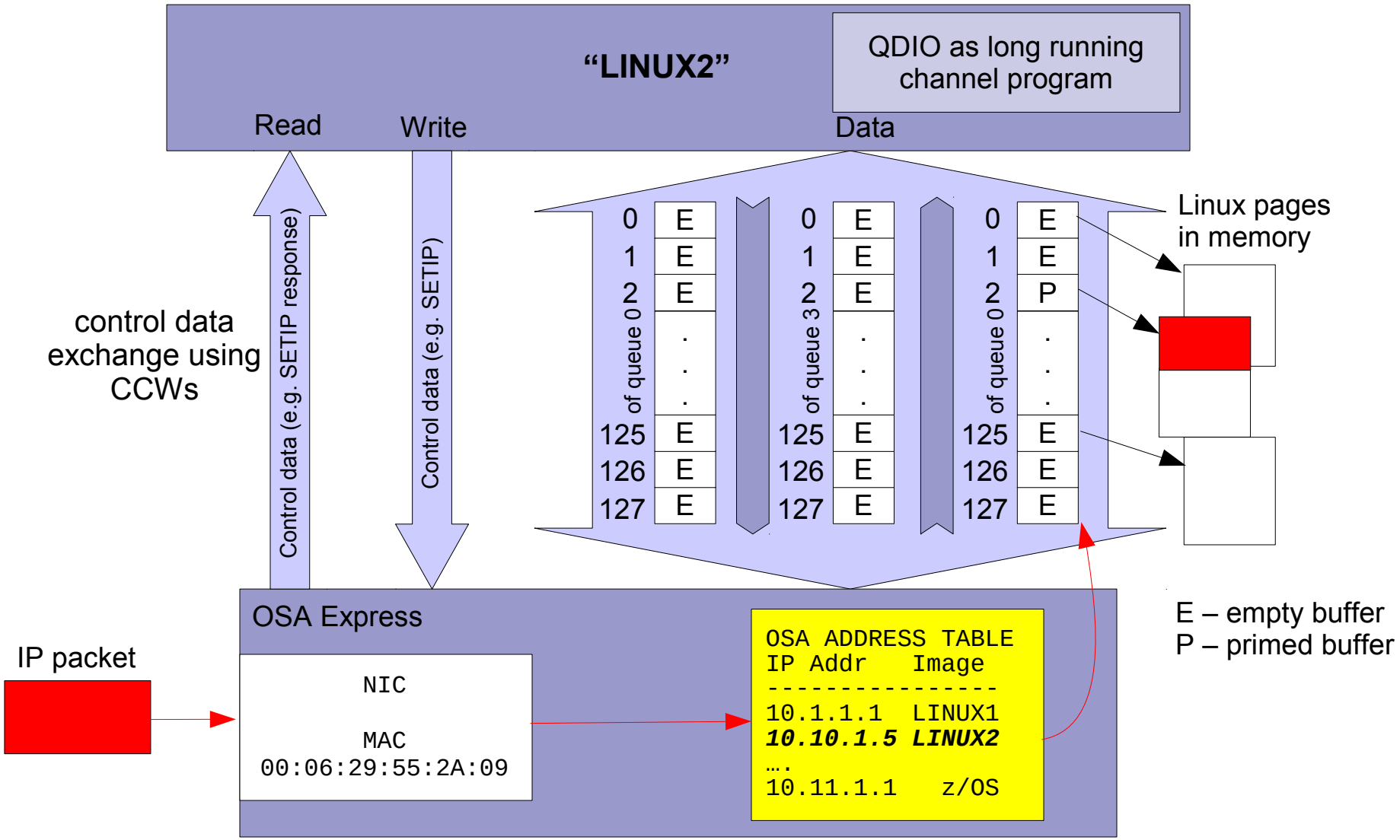
- Special purpose GuestLAN
  - Ethernet, type QDIO
  - Built-in IEEE 802.1q bridge to outside network
- 1-8 associated OSA-connections
- Restricted
- Persistent

- Failover and Link Aggregation
- Port Isolation

- **Virtual Network Devices – NICs (Virtual Network Interface Cards)**

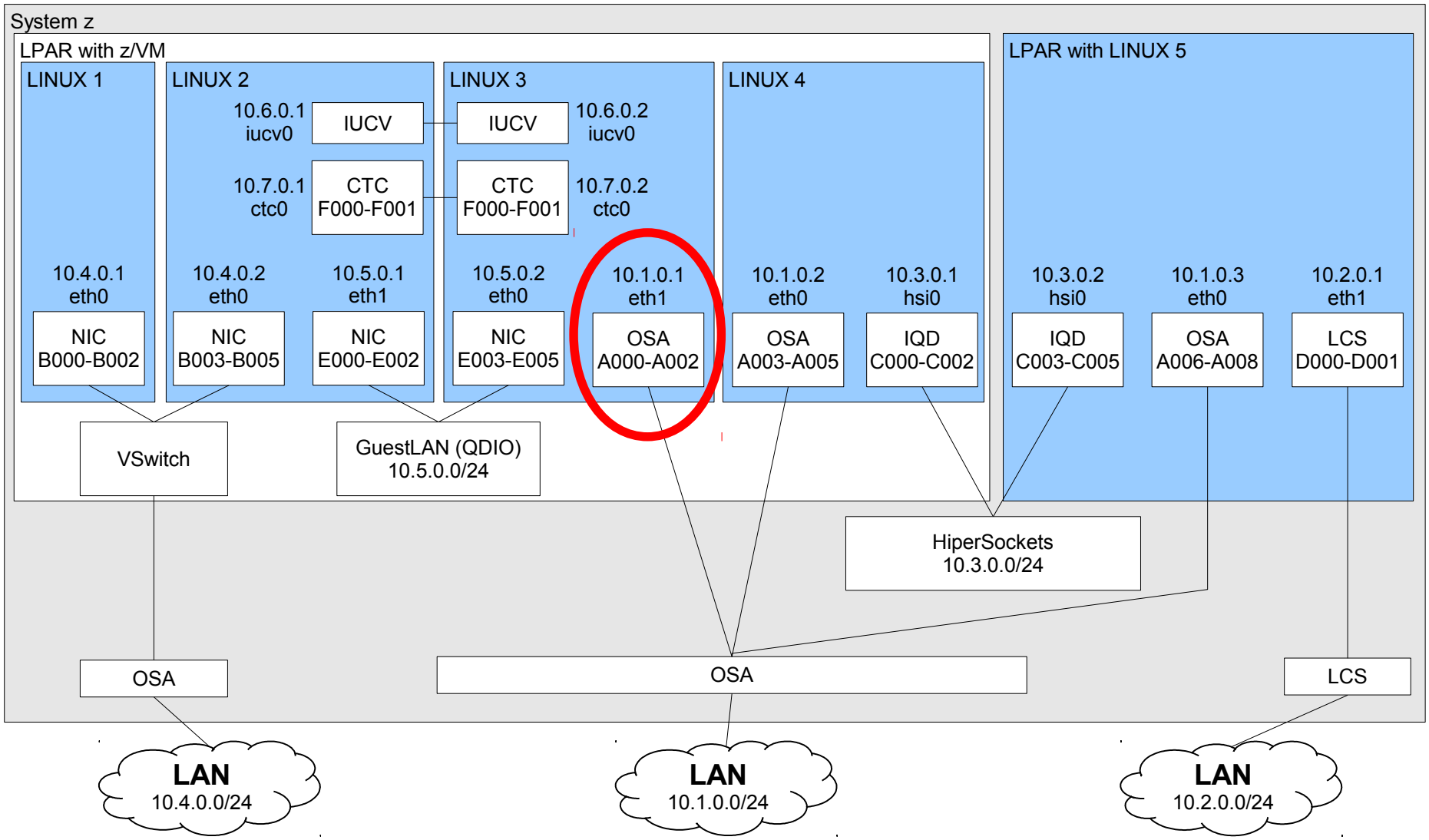
- Defined by directory or CP DEFINE NIC command
  - Type QDIO or HIPERS (must match LAN type)
- The only thing visible to Linux

The Queued Direct I/O (QDIO) Architecture



Configuration

Networking Device Configuration - Example



Network Device Configuration - Generic

Load the device driver module:

```
# modprobe qeth
```

Create a new device by grouping its CCW devices:

```
# echo 0.0.a000,0.0.a001,0.0.a002 >/sys/bus/ccwgroup/drivers/qeth/group
```

Set optional attributes

```
# echo 32 > /sys/devices/qeth/0.0.a000/buffer_count
```

Set the device online

```
# echo 1 > /sys/devices/qeth/0.0.a000/online
```

automatically assignes an interface name to the qeth device:

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

Configure an IP address

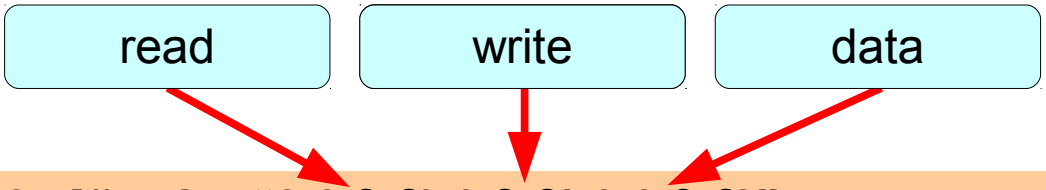
```
# ifconfig eth0 10.1.0.1 netmask 255.255.255.0
```

Configuration of the qeth driver (cont.)

- The qeth device driver automatically assigns interface names to the qeth group device and creates the corresponding sysfs structures
- The following name schema is used:
  - eth[n] for ethernet
  - hsi[n] for Hipersocket devices
  - tr[n] for Token Ring
  - osn[n] for ESCON bridge

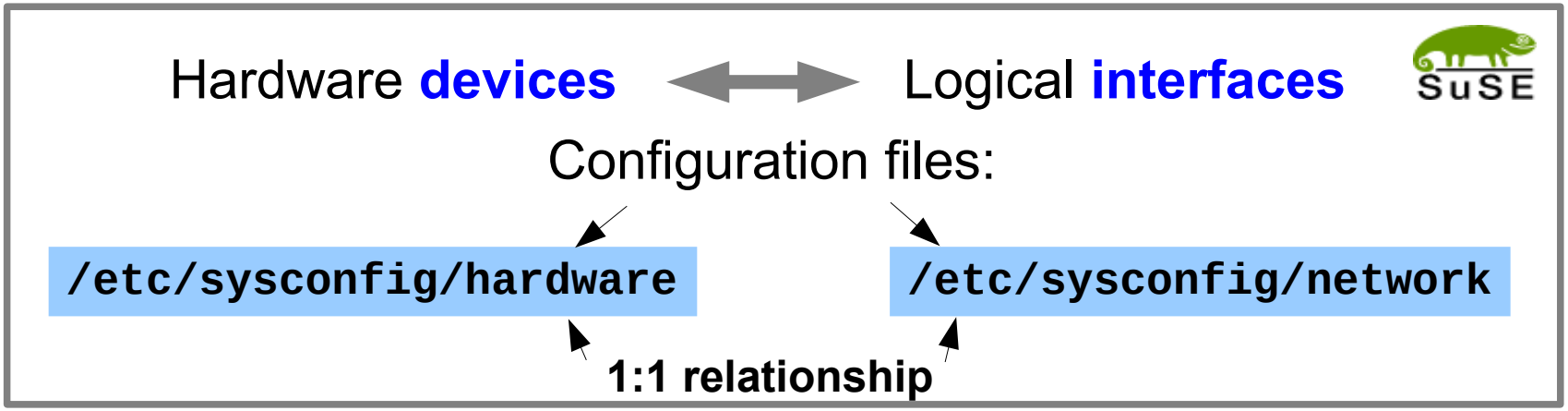
The qeth device driver shares the name space for Ethernet and Token Ring interfaces with the LCS device driver!

```
[root@t6345040 ~]# modprobe qeth
```




```
[root@t6345040 ~]# echo “0.0.f5f0,0.f5f1,0.0.f5f2” >
/sys/bus/ccwgroup/drivers/qeth/group
```

SuSE SLES 10 Network Configuration



Naming convention:

```
hw/ifcfg-<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>
    /usr/share/doc/packages/sysconfig/README and README.s390
```

Static QETH Device Setup (SuSE SLES 10)

For LINUX1 eth0

1. Create a 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'
```

further attributes

Static QETH Device Setup (SuSE SLES 10) (cont.)

2. Create an interface configuration file:

```
/etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.a000
   BOOTPROTO='static'
   BROADCAST='10.1.255.255'
   IPADDR='10.1.1.1'
   NETMASK='255.255.0.0'
   NETWORK='10.1.0.0'
   STARTMODE='onboot'
```

===> hardware device always gets the right IP address

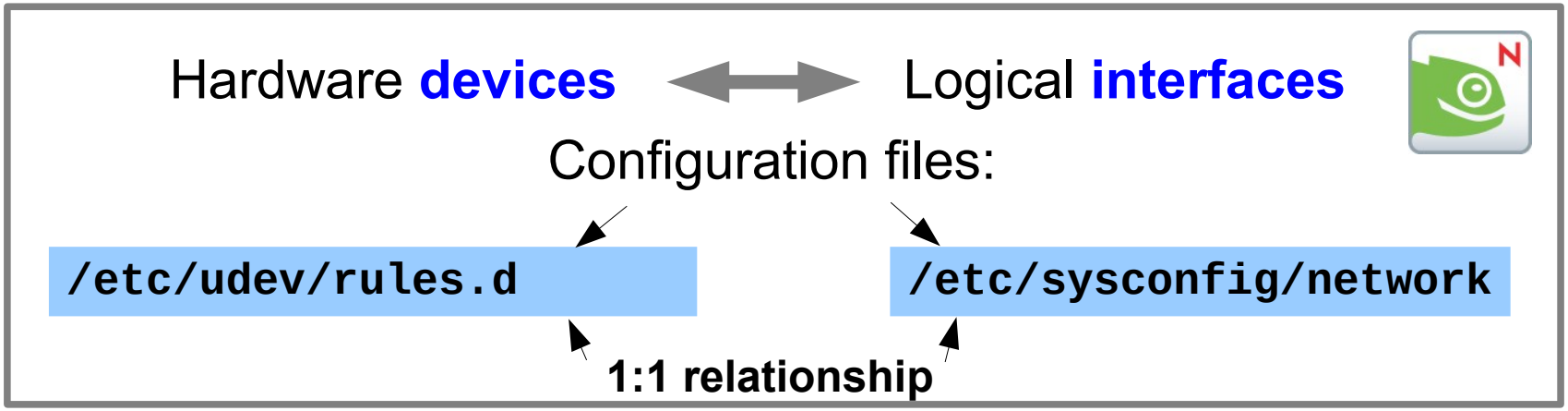
Explanations are found in

```
/etc/sysconfig/network/ifcfg.template
```

3. Before reboot: test your config files:

```
#> hwup qeth-bus-ccw-0.0.a000
```

SuSE SLES 11 Network Configuration



- Devices are configured via udev (framework for dynamic device conf.)
- udev naming rules: `51-<device type>-<bus location>.rules`
  e.g. `51-qeth-0.0.a000.rules`
- Persistent naming: Mapping bus ↔ interface with udev rule
  `70-persistent-net.rules`
- Interface naming convention: `ifcfg-<ifname>`, e.g. `ifcfg-eth0`
- Scripts: `qeth_configure` and `ifup/ifdown`

Static QETH Device Setup (SuSE SLES 11)

For LINUX1 eth0

Created by Yast or
qeth_configure -1 0 0.0.a000 0.0.a001 0.0.a002 1

1. Hardware Device Configuration rule:

```
/etc/udev/rules.d/51-qeth-0.0.a000.rules:
 # configure qeth device at 0.0.a000/0.0.a001/0.0.a002
 ...
 ACTION=="add", SUBSYSTEM=="ccwgroup",
        KERNEL=="0.0.a000", ATTR{layer2}="0"
```

further attributes

2. Entry in persistenet naming rule:

```
/etc/udev/rules.d/70-persistent-net.rules:
 SUBSYSTEM=="net", ACTION=="add", DRIVERS=="qeth",
        KERNEL=="0.0.a000", ATTR{type}=="1",
        KERNEL=="eth*", NAME="eth1"
```

RedHat RHEL5 Network Configuration

- Configuration files:

/etc/modprobe.conf

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

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

```
NETTYPE       qeth | lcs | ctc | iucv
TYPE          Ethernet | CTC | IUCV
SUBCHANNELS   0.0.b003,0.0.b004,0.0.b005
PORTNAME
OPTIONS
MACADDR
```

- `ifup/ifdown` scripts contain mainframe-specifics

Static QETH Device Setup (RedHat RHEL5)

For LINUX 1 eth0

1. Create the 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=10.1.255.255
  IPADDR=10.1.1.1
  NETMASK=255.255.0.0
  OPTIONS='checksumming=hw_checksumming'
```

further attributes

Static QETH Device Setup (RedHat RHEL5) (cont.)

redhat

2. Add or verify alias in /etc/modprobe.conf:

```
/etc/modprobe.conf:
   ...
   alias eth0 qeth
   ...
```

3. For details see:

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

Network Device Drivers – Advanced

QETH Device sysfs Attribute checksumming

- additional redundancy check to protect data integrity
- Offload checksumming for incoming IP packages from Linux network stack to OSA-card

  `QETH_OPTIONS='checksumming=hw_checksumming'` or

```
#> echo hw_checksumming >
  /sys/devices/qeth/0.0.b004/checksumming
```

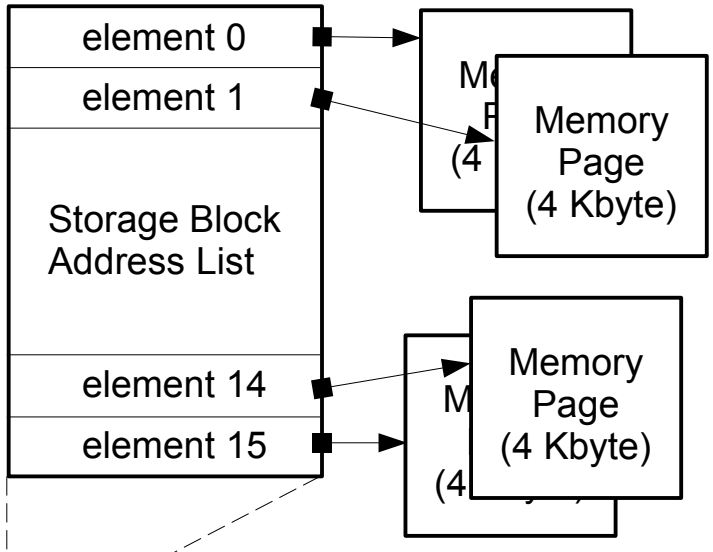
  ===> move workload from Linux to OSA-Express adapter
- Available for OSA-devices in layer3 mode only
- for trused HiperSockets devices:

  `QETH_OPTIONS='checksumming=hw_checksumming'` or

```
#> echo no_checksumming >
  /sys/devices/qeth/0.0.b004/checksumming
```

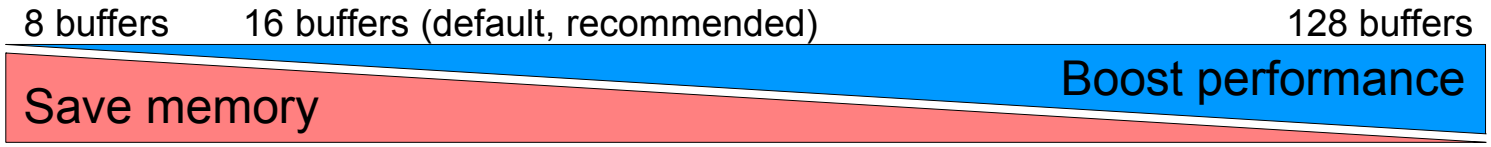
QETH Device sysfs Attribute `buffer_count`

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



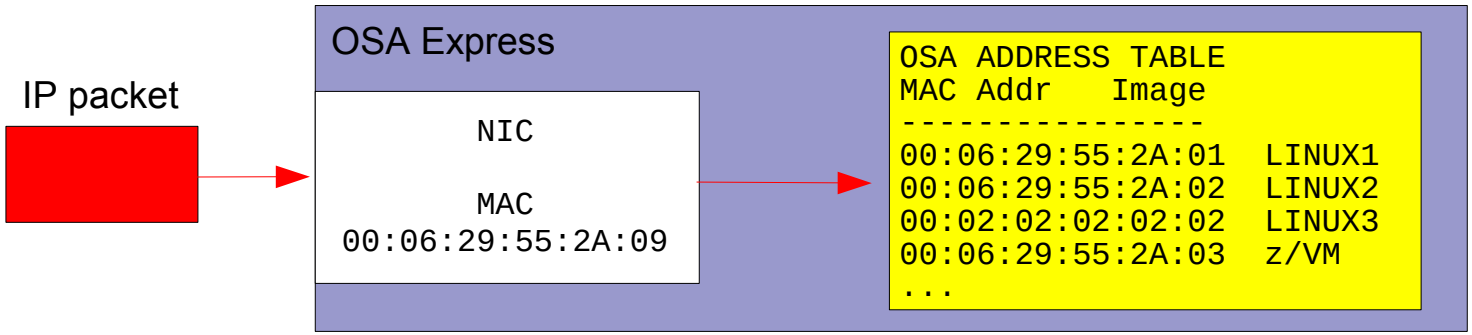
Per QETH card memory usage:

control data structures:	~ 200 KB
memory for one buffer:	64 KB
`buffer_count = 8`	--> ~ 712 KB
`buffer_count = 128`	--> ~ 8.4 MB



QETH Layer 2 mode

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



- udev rule 51-qeth-... (SLES11): **`...ATTR{layer2}=1`**
- or command (SLES11): **`qeth_configure -l 1 ...`**
- hwcfg-qeth... file (SLES10): **`QETH_LAYER2_SUPPORT=1`**
- ifcfg-qeth... file (SLES10): **`LLADDR='<MAC Address>'`**
- ifcfg-... file (RHEL5): **`MACADDR='<MAC Address>'`**
  `OPTIONS='layer2=1'`
- Direct attached OSA: MAC address must be defined manually
  `ifconfig eth0 hw ether 00:06:29:55:2A:01`
- HiperSocket: **new** layer2 support starting with z10
  MAC address automatically generated
- VSWITCH or GuestLAN under z/VM: MAC address created by z/VM

QETH Layer 2 mode (cont.)

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

- activating Layer 2 is done per device via sysfs attribute
- possible layer 2 values:
  - 0: use device in layer 3 mode
  - 1: use device in layer 2 mode

- Setting of layer2 attribute is only permitted when device is offline
- Advantages:
  - Independent of IP-protocol
  - DHCP, tcpdump work without option fake_ll
  - Channel bonding is possible
  - No OSA-specific setup is necessary for
    - Routing, IP Address Takeover, Proxy ARP

QETH Layer 2 mode (cont.)

- Direct attached OSA
  - Restriction: Older OSA-generation (<= z990): Layer2 and Layer3 traffic can be transmitted over the same OSA CHPID, but not between two hosts sharing the same CHPID !
- HiperSocket (new with z10)
  - Layer2 and Layer3 traffic separated
- GuestLAN type QDIO supported
  - GuestLAN definition for layer2:

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

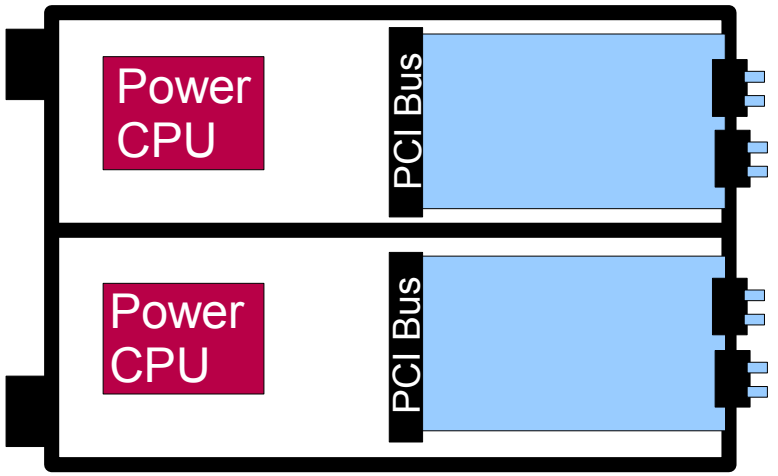
- VSWITCH

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

OSA Express 3 – 2 ports within one CHPID

- OSA Express2 – 2 CHPIDs with 1 port per CHPID – 2 ports totally
- OSA Express3 – 2 CHPIDs with 2 ports per CHPID – 4 ports totally (z10)
- New sysfs-attribute “portno” can contain '0' or '1'
- OSA-Express3 GbE SX and LX on z10
- udev rule 51-qeth-... (SLES11):
  or command (SLES11):
  `...ATTR{portno}=1`
  `qeth_configure -n 1 ...`
- hwcfg-qeth... file (SLES10 SP2) : `QETH_OPTIONS=”portno=1”`
- ifcfg-... file (RHEL5.2): `OPTIONS='portno=1'`

- Provides Hardware data router function
  ==> reduced latency
  ==> full linespeed achieved



Commands and tools for qeth-driven devices

- List of known qeth devices: cat /proc/qeth or lsqeth -p

```
#> 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.b000/0.0.b001/0.0.b002 xB0   hsi0       HiperSockets 0    sw
```

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

```
#> lsqeth eth0
Device name                          : eth0
-------------------------------------------
        card_type                    : OSD_1000
        cdev0                        : 0.0.f5f0
        cdev1                        : 0.0.f5f1
        cdev2                        : 0.0.f5f2
        chpid                        : 76
        online                       : 1
        checksumming                 : sw checksumming
        state                        : UP (LAN ONLINE)
        buffer_count                 : 16
        layer2                       : 0
```

Clip of displayed attributes only

Commands and tools for qeth-driven devices (cont.)

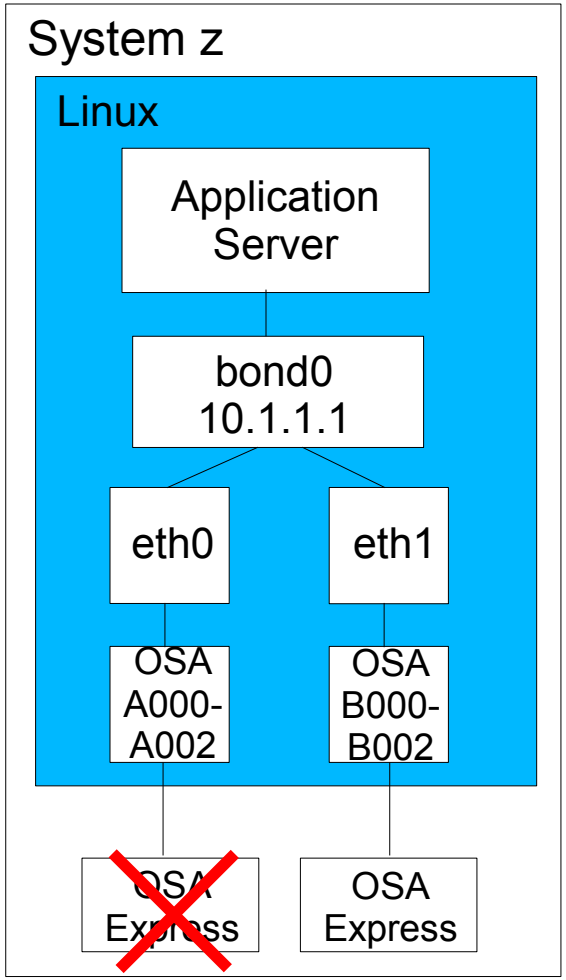
- Managing IP-addresses on OSA / HiperSockets: **`qetharp`**
  - Suitable for layer3 devices only
- Configuration support for IPA, VIPA, Proxy ARP: **`qethconf`**
  - Suitable for layer3 devices only

Other networking tools for System z

- SNMP support: **`osasnmpd`**
  - Subagent for the snmpd daemon to provide OSA Express information
- Linux image control for LPAR and z/VM: **`snipl`**
  - Can boot, stop, reset Linux images, send and receive OS messages

Channel Bonding

- The Linux bonding driver provides a method for aggregating multiple network interfaces into a single logical “bonded” interface
- provides failover and / or load-balancing active / backup /aggregation modes
- Detects loss of NIC connectivity
  ==> automatic failover
- transparent for LAN infrastructure
- applies to layer2-devices only
- no dynamic routing (OSPF) dependency
- latest setup description:
  http://sourceforge.net/projects/bonding/



Channel bonding setup

- Add MAC address to eth0 & eth1
  (not necessary for GuestLAN or Vswitch)

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

- Load bonding module with miimon option
  (Otherwise bonding will not detect link failures)

```
#> modprobe bonding miimon=100 mode=balance-rr
```

- Bring up bonding device bond0

```
#> ifconfig bond0 10.1.1.1 netmask 255.255.255.0
```

- Connect eth0 & eth1 to bond0

```
#> ifenslave bond0 eth0
#> ifenslave bond0 eth1
```

Channel bonding setup (SLES10 – config files)

- Interface configuration file for a slave

```
/etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.a000
   BOOTPROTO='static'
   IPADDR=''
   SLAVE='yes'
   STARTMODE='onboot'
```

- Interface configuration file for a master

```
/etc/sysconfig/network/ifcfg-bond0
   BOOTPROTO='static'
   BROADCAST='10.1.255.255'
   IPADDR='10.1.1.1'
   NETMASK='255.255.0.0'
   NETWORK='10.1.0.0'
   STARTMODE='onboot'

   BONDING_MASTER='yes'
   BONDING_MODULE_OPTS='mode=1 miimon=1'
   BONDING_SLAVE0='qeth-bus-ccw-0.0.a000'
   BONDING_SLAVE1='qeth-bus-ccw-0.0.b000'
```

Channel bonding setup (RHEL5 – config files)

- interface configuration file for slave

```
/etc/sysconfig/network/ifcfg-eth0
        DEVICE=eth0
        IPADDR=''
        SLAVE='yes'
        MASTER='bond0'
```

- interface configuration file for master

```
/etc/sysconfig/network/ifcfg-bond0
        DEVICE=bond0
        BROADCAST='10.1.255.255'
        IPADDR='10.1.1.1'
        NETMASK='255.255.0.0'
        NETWORK='10.1.0.0'
```

- Module loader

```
/etc/modprobe.conf
        alias eth0 qeth
        alias eth1 qeth
        alias bond0 bonding
        options bond0 miimon=100 mode=1
```

Channel bonding setup (cont.)

```
#> 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:01
          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500 ...
```

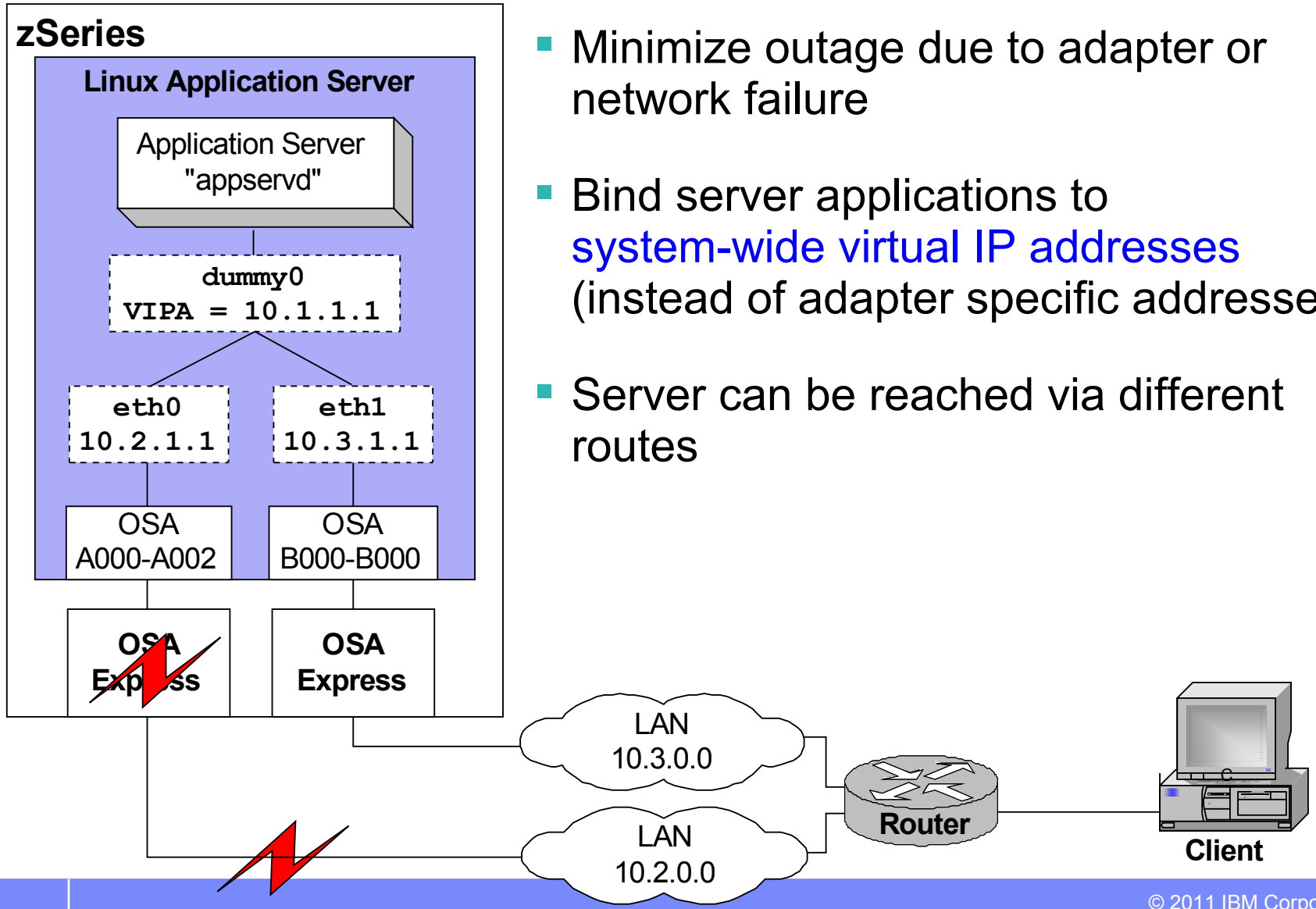
```
#> cat /proc/net/bonding/bond0

Bonding Mode: load balancing (round-robin)
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: 00:05:27:54:21:04
```

Virtual IP addresses



- Minimize outage due to adapter or network failure
- Bind server applications to system-wide virtual IP addresses (instead of adapter specific addresses)
- Server can be reached via different routes

Virtual IP Address Setup

- Create a virtual interface and assign the VIPA using a dummy interface

```
#> modprobe dummy
#> ifconfig dummy0 10.1.1.1 netmask 255.255.0.0
```

  or using an interface alias

```
#> ifconfig eth0:1 10.1.1.1 netmask 255.255.0.0
```

- Layer 3 only: register virtual IP address with physical devices

```
#> echo 10.1.1.1 > /sys/class/net/eth0/device/vipa/add4
#> echo 10.1.1.1 > /sys/class/net/eth1/device/vipa/add4
```

- On the router add a route to the routing table

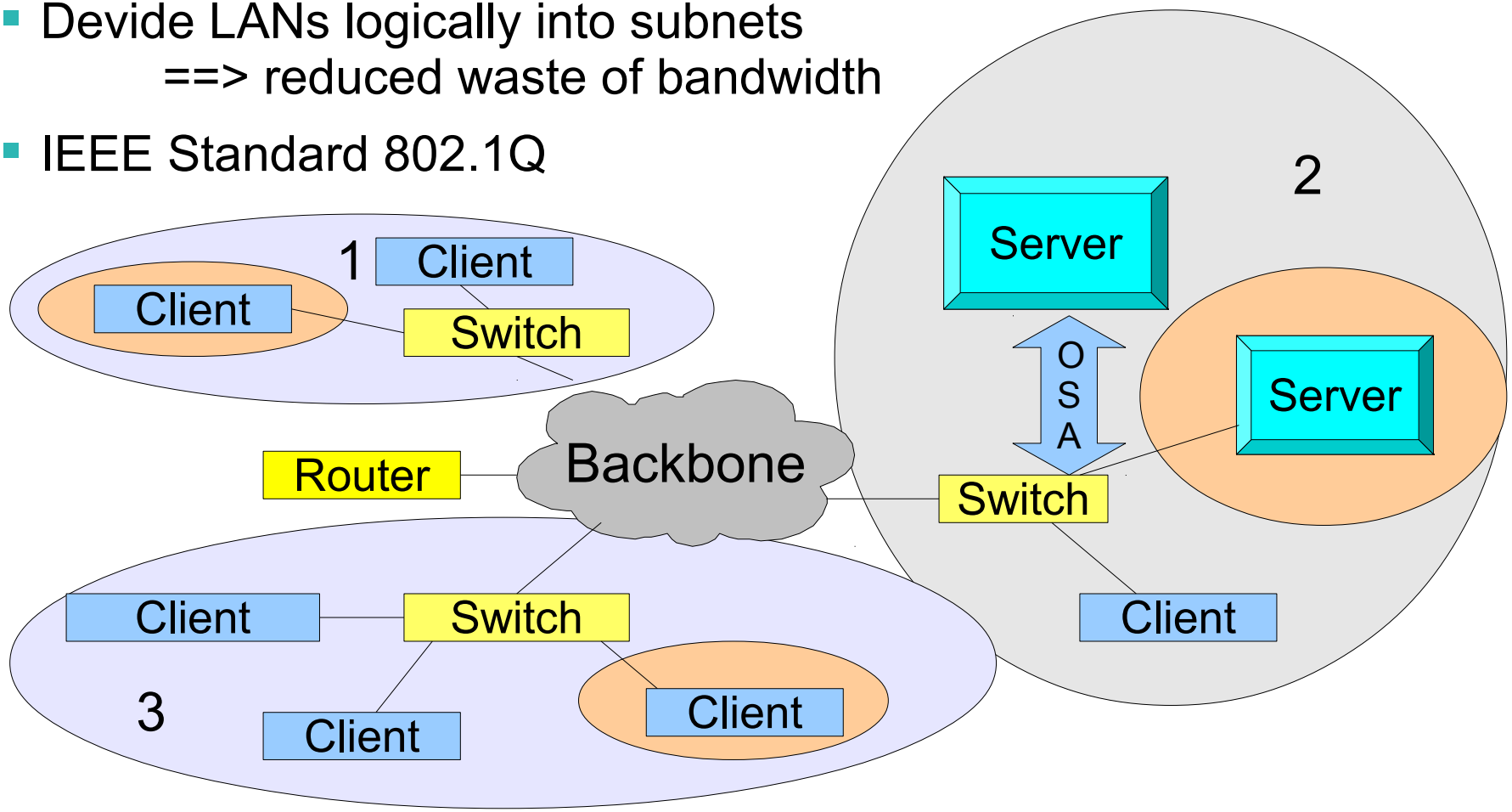
```
#> route add -host 10.1.1.1 gw 10.2.1.1
#> route add -host 10.1.1.1 gw 10.3.1.1
```

if LAN1 works
if LAN2 works

or, better, configure the routes with a dynamic routing daemon (e.g. quagga: http://quagga.net).

Virtual LAN (VLAN) support

- Risk of big switched LANs: flood of broadcast traffic
- Devide LANs logically into subnets
  ==> reduced waste of bandwidth
- IEEE Standard 802.1Q



Virtual LAN (VLAN) support (cont.)

- Setup

```
#> ifconfig eth1 9.164.160.23 netmask 255.255.224.0
#> vconfig add eth1 3
#> ifconfig eth1.3 1.2.3.4 netmask 255.255.0.0
```

- Display information

```
#> cat /proc/net/vlan/config
   VLAN Dev name    | VLAN_ID
   Name-Type: VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD
   eth1.3           | 3  | eth1
```

- Implemented
  VLAN tag, added to packets transmitted
- Supported by
  real OSA-card, HiperSockets, z/VM Guest LAN, z/VM VSWITCH

Interface names

Interface Name	Device Driver	Interface / Link Type	Model / Submodel	Used for
eth<x>	qeth lcs lcs	Ethernet	1731/01 3088/01 3088/60	OSA-card / type OSD P390-LCS-card OSA-card / type OSE
hsi<x>	qeth	Ethernet	1731/05	HiperSockets / type IQD
tr<x>	qeth lcs lcs	Token Ring	1731/01 3088/01 3088/60	OSA-card / type OSD P390-LCS-card OSA-card / type OSE
osn<x>	qeth	SNA<->Ethernet	1731/06	OSA-card / type OSN
ctc<x>	ctc	Point-to-Point	3088/08 3088/1e 3088/1f virtual	Channel-To-Channel adapter FICON adapter ESCON adapter VM-guest communication
iucv<x>	netiucv	Point-to-Point	virtual	VM-guest communication

Summary of Linux Network Device Drivers

	QETH					LCS	CTC	IUCV
	OSA	Hiper-Sockets	GuestLAN QDIO	GuestLAN Hiper	VSWITCH			
Adapters	100 Mbps, 1/10Gbps, 1000 Base-T, HSTR					100 Mbps, 1000 Base-T, HSTR	ESCON, FICON, Virtual CTC/A	
Connection type	LAN	LAN	LAN	LAN	LAN	LAN	point-to-point	point-to-point
Layer	Layer2 / 3	Layer2 /3	Layer2 / 3	Layer3	Layer2 / 3	Layer3		
Protocols	IPv4, IPv6	IPv4, Ipv6	IPv4, IPv6	IPv4		IPv4	IPv4	IPv4
Remarks	**Primary network device driver for Linux on System z**					restricted access (admin defines OSA Address Table)	Deprecated	Deprecated

AF_IUCV protocol support

- Enable socket applications in Linux to use the Inter-User Communication Vehicle (IUCV) in z/VM
- Communication between z/VM guests
- Stream-oriented sockets (SOCK_STREAM) and
- Connection-oriented datagram sockets (SOCK_SEQPACKET)
- SLES9 SP4, SLES10 SP2, RHEL5 U2, and SLES11 (module af_iucv)

```
struct sockaddr_iucv {
  sa_family_t     siucv_family;     /* 32           */
  unsigned short  siucv_port;       /* Reserved     */
  unsigned int    siucv_addr;       /* Reserved     */
  char            siucv_nodeid[8];  /* Reserved     */
  char            siucv_userid[8];  /* Guest UserId */
  char            siucv_name[8];    /* Appl. Name   */
```

AF_IUCV socket calls

- Calls to establish connection

```
sockno = socket(32, SOCK_STREAM, 0)
bind(sockno, own_iucv_sockaddr, len)
listen(sockno, backlog)
accept(sockno, client_iucv_sockaddr, len)
connect(sockno, server_iucv_sockaddr, len)
```

- Transfer calls

```
read / write, recv / send
```

- Finishing calls

```
shutdown / close
```

References

- Linux on System z on developerWorks
  http://www.ibm.com/developerworks/linux/linux390/index.html
- Linux on System z Documentation
  http://www.ibm.com/developerworks/linux/linux390/development_documentation.html
- Linux on System z, snipl
  http://www.ibm.com/developerworks/linux/linux390/snipl.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
- IBM System z OSA-Express Implementation Guide
  http://www.redbooks.ibm.com/abstracts/sg245948.html

Questions?

Martin Schwidefsky

Linux on System z Development

Schönaicher Strasse 220
71032 Böblingen, Germany

Phone +49 (0)7031-16-2247
schwidefsky@de.ibm.com