Welcome to the Jungle: Linux on IBM Z Networking

—

Stefan Raspl
Linux on IBM Z Development

Trademarks

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

AIX*	DB2*	HiperSockets*	MQSeries*	PowerHA*	RMF	System z*	zEnterprise*	z/VM*
BladeCenter*	DFSMS	HyperSwap	NetView*	PR/SM	Smarter Planet*	System z10*	z10	z/VSE*
CICS*	EASY Tier	IMS	OMEGAMON*	PureSystems	Storwize*	Tivoli*	z10 EC	
Cognos*	FICON*	InfiniBand*	Parallel Sysplex*	Rational*	System Storage*	WebSphere*	z/OS*	
DataPower*	GDPS*	Lotus*	POWER7*	RACF*	System x*	XIV*		

* 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 therefrom.
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.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
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.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and
Linux is a registered trademark of Linus Torvalds 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.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.

* Other product and service names might be trademarks of IBM or other 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.

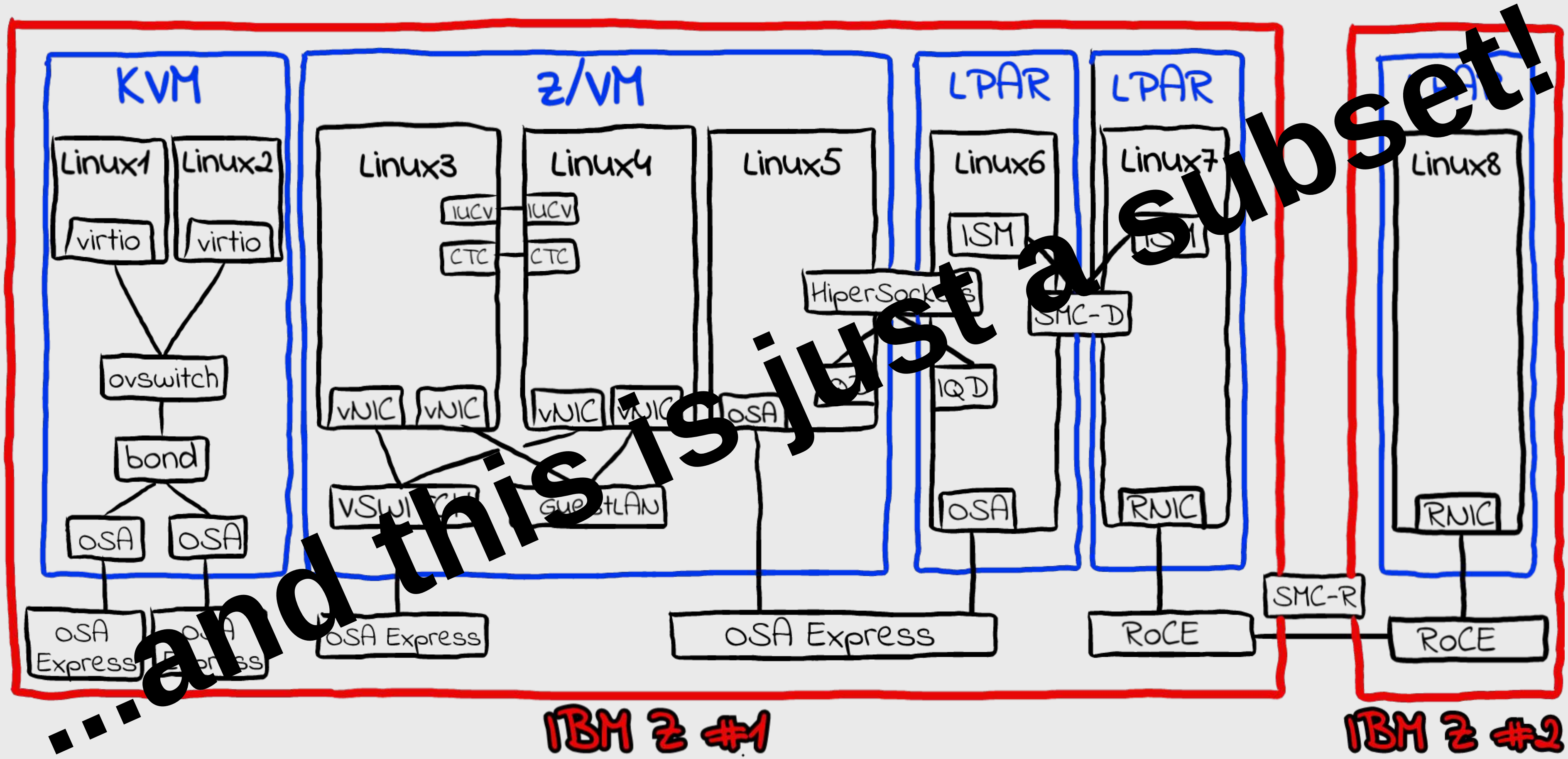
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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.

This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g, zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Networking Options for Linux on Z (selection)



Agenda

- Introduction
- Part I: Common Linux on Z Networking Facilities
  - Networking Cards
    - OSA-Express & RoCE Express
    - Shared Devices Traffic
  - Channel Bonding
  - HiperSockets
  - Shared Memory Communications
    - SMC-D
    - SMC-R
- Part II: Environment-specific Networking Facilities and Considerations
- References

OSA-Express

- Most recent models:
  - ***OSA-Express7S***: 25GbE
  - ***OSA-Express6s***: 10 GbE, 1GbE and 1000Base-T
- 1, 10 and 25GbE models with varying HW features:
  - 1GbE: Base-T or fiber optics, 2 ports
  - 10 and 25GbE: Fiber only, 1 port
- 25GbE model strictly requires 25GbE capable switch – no negotiation to 10GbE
- Considered platform's native networking card
- Supported by all operating systems on IBM Z
- Supports TCP/IP[1] traffic only
- Up to 480 IP stacks per port and 48 cards in an IBM z14

RoCE Express

- Most recent models:
  - ***RoCE Express2***: 25GbE and 10GbE (Fiber optics only)
- Introduced with zEC12 for SMC-R
- 10 and 25GbE models, optical connectors only
- 25GbE model strictly requires 25GbE capable switch – no negotiation to 10GbE
- All models feature 2 ports
- TCP/IP[1] or RoCE (RDMA over Converged Ethernet)
- TCP/IP functionality exploited by Linux only
- Up to 63 IP stacks per port and 8 cards in an IBM z14

[1] *Synonymous to any kind of "traditional" network traffic (UDP, SCTP, et al)*

OSA-Express

- **Features** (selection)
  - HW offloads: Checksumming, TCP segmentation offload (***TSO***)
  - Layer 2 and layer 3 mode
  - VLAN, QoS, VIPA, ARP, et al
- **RAS**
  - Extended RAS
  - Concurrent firmware updates
    95+ percent completely concurrent
- **Layer modes supported**
  - ***Layer 2*** (**default**, recommended): Maximum compatibility with Linux tooling and frameworks
  - ***Layer 3***: Reduced compatibility.
    OSA handles ARP, special support for VIPA, Proxy ARP, IP Address Takeover.

RoCE Express

- **Features** (selection)
  - HW offloads: Checksumming, TSO
  - RDMA over Converged Ethernet (RoCE)
  - Flow Control, Explicit Congestion Notification
  - IPoIB, uDAPL, et al
  - VLAN, QoS, et al
- **RAS**
  - Regular RAS
  - Changing optics of a single card disrupts entire PCHID
  - Firmware updates are disruptive
- **Layer modes supported**
  - Layer 2 only

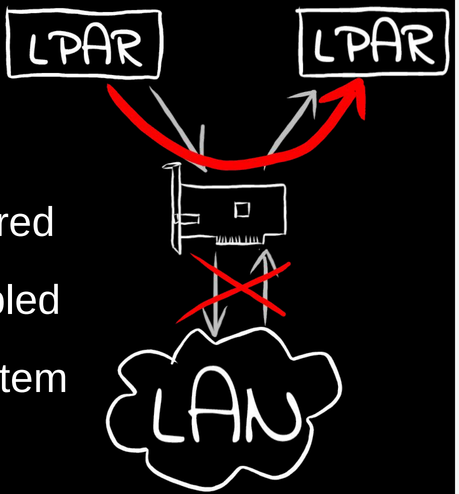
OSA-Express

- **CCW group device**
  Consists of three device numbers:
  - *Read* device (control data ⇐ OSA)
  - *Write* device (control data ⇒ OSA)
  - *Data* device (network traffic)
- **Physical identifier**: Card identified by PCHID
- **Device Drivers**:
  - **`qeth`**: Covers all OSA-Express models (and HiperSockets) in QDIO mode
  - **`lcs`** (alternative driver):
    - OSE CHPIDs
    - IP address must be set in OSA/SF
    - Utilizes regular CCW instead of QDIO mode ⇒ inferior performance

RoCE Express

- Regular **PCI device**
- **Physical Identifier**:
  - RoCE Express: FID identifies card
  - RoCE Express2: FID identifies port
- **Device Drivers**:
  - **`mlx4`**: RoCE Express
  - **`mlx5`**: RoCE Express2

OSA-Express



- Shortcut within device
- No extra configuration required
- Will not work with TSO enabled
- Works with all operating system images on Z
- **Controlling shared traffic**:
  - VEPA (*Virtual Edge Port Aggregator*) mode: Send all traffic to adjacent switch for consistent enforcement of security policy. Requires reflective relay mode in switch!
  - Alternative: Drop any traffic intended for other OS image sharing the same OSA device

RoCE Express

- Excellent throughput
- Shared TCP/IP traffic works with Linux images only due to lack of support in other operating systems. I.e. no shared Ethernet traffic with
  - z/OS
  - z/VSE
  - z/VM
- Shared RDMA traffic (SMC-R) with z/OS works
- No controls for control shared traffic

OSA-Express

- **When to use**
  - Vast virtualization capabilities required
  - Economic CPU usage
  - Excellent RAS capabilities
  - z/VM VSWITCH external connectivity
  - Shared Device: Saves CPU cycles (as compared to HiperSockets)
  - LCS: Security aspects at cost of performance

RoCE Express

- **When to use**
  - Very low latency
  - Implement SMC-R with a single device
  - 2 Ports on all models
  - Shared Device: Excellent throughput

OSA-Express

- **What to consider**
  - Limited shared network traffic
  - Shared network traffic without TSO only
- **z/OS Connectivity**

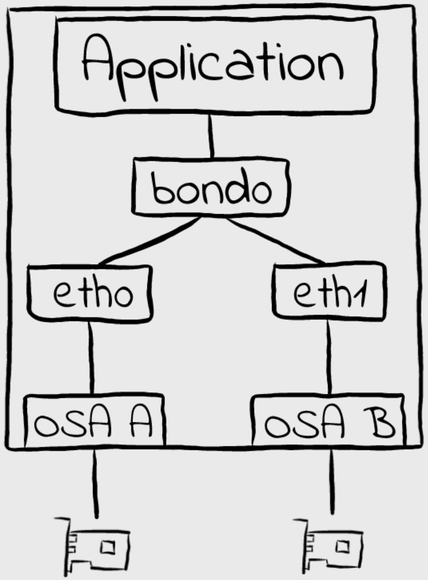
  No limitations

RoCE Express

- **What to consider**
  - Limited virtualization capabilities
  - Limited plugging capacity
  - Regular RAS only
  - Can result in higher CPU consumption (as compared to OSA)
  - Not supported by z/VM VSWITCH and Open vSwitch
- **z/OS Connectivity**
  - RoCE Express supported for RDMA traffic (⇒SMC-R) only
  - z/OS requires OSA devices for external connectivity
  - no shared network traffic Linux ⇔ z/OS for non-RDMA

Linux `bonding` Driver

- Use Linux **`bonding`** driver to aggregate multiple network interfaces into a single logical “bonded” interface
- Recommended driver for channel bonding
- Works with both, OSA-Express and RoCE Express cards
  - However: OSA devices in layer 2 mode only!
- Various modes available, providing HA or load-balancing functionality
  Note: LACP (*Link Aggregation Control Protocol*, see IEEE 802.3ad) requires *dedicated* ports
- See white paper *Linux Channel Bonding Best Practices and Recommendations* at https://ibm.biz/BdzMsJ for further details



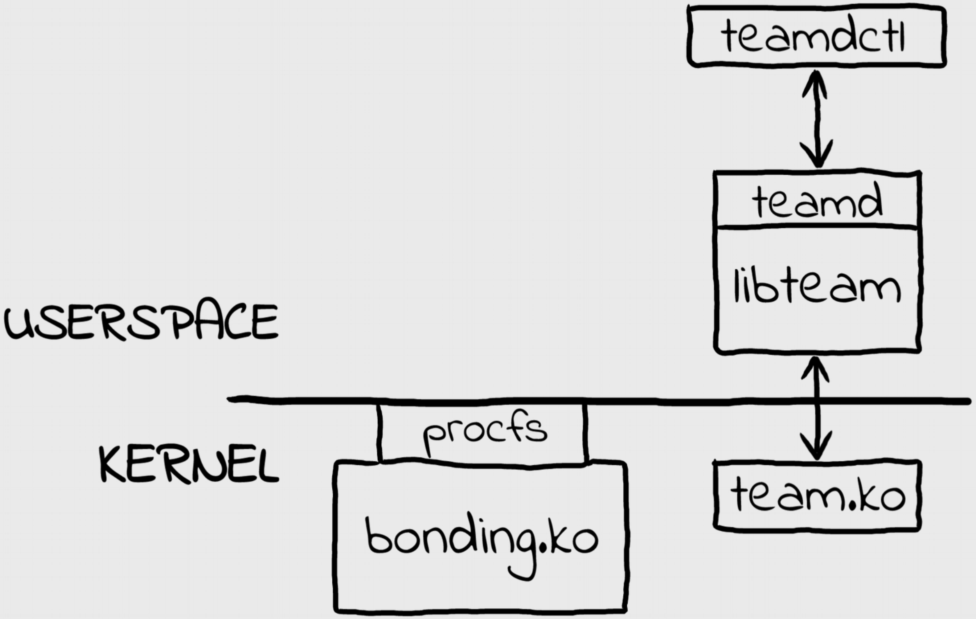

```
# load bonding module with miimon
# option (enables link monitoring)
$ modprobe bonding miimon=100 mode=balance-rr

# add MAC addresses to slave devices eth0 & eth1
# (not necessary for VSWITCH)
$ ip link set dev eth0 address 00:06:29:55:2A:01
$ ip link set dev eth1 address 00:05:27:54:21:04

# activate the bonding device bond0
$ ip addr add 10.1.1.1/24 dev bond0

# connect slave devices eth0 & eth1 to
# bonding device bond0
$ ifenslave bond0 eth0 eth1
```

Teaming Driver



- Alternative to Linux kernel's "*bonding*" module: "*Solve the same problem using a different approach*"
  ⇒ *comparable functionality*
- Works with both, OSA-Express and RoCE Express cards
  - OSA: Layer 2 devices only
- Various modes available, providing HA or load-balancing functionality
  Note: LACP (*Link Aggregation Control Protocol*, see IEEE 802.3ad) requires *dedicated* ports
- Different architecture, relying on userspace components
- Different terminology as compared to bonding driver:
  - "*team*" vs "*bond*" device
  - "*ports*" vs "*slaves*"
  - "*runners*" vs "*bonding modes*"
- Various programming APIs
- See http://libteam.org/ for further details

```
# start teaming daemon in background,
# creates instance team0 in round-robin mode
$ teamd -d

# add ports (=slaves)
$ teamdctl team0 port add eth1
$ teamdctl team0 port add eth2

# add IP address and activate
$ ip addr add 192.168.3.37 dev team0
$ ip link set team0 up
```

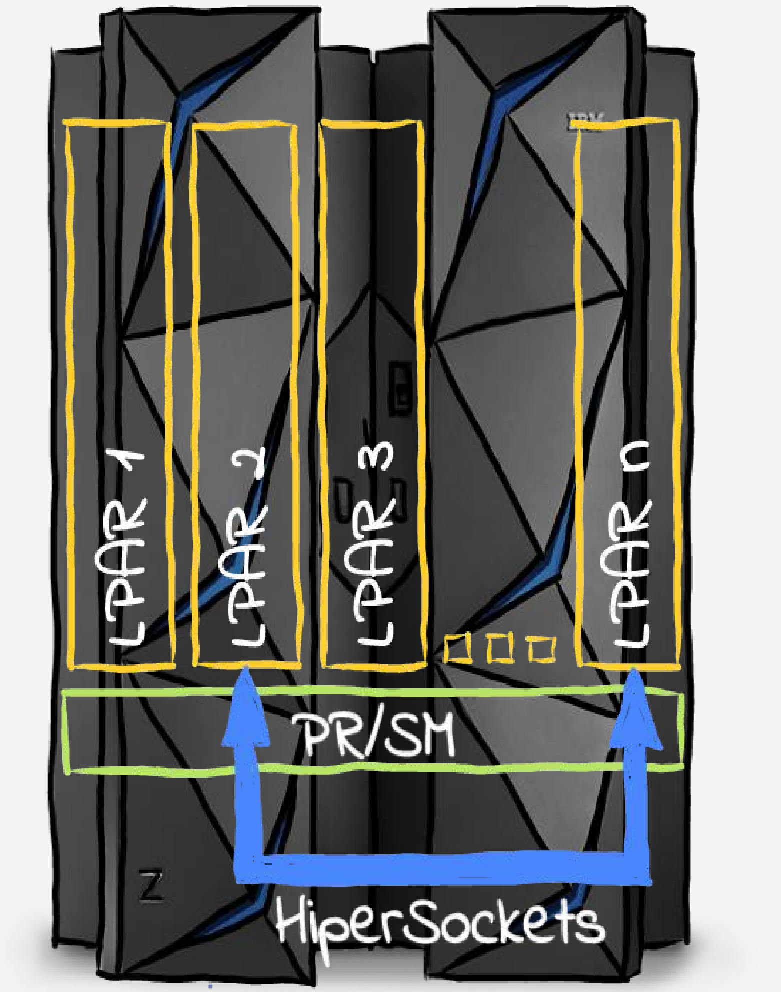
Summary

- **When to use**
  - High availability
  - Increased throughput

- **What to consider**
  - `bonding` driver is recommended
  - Consult the following whitepaper for specifics on recommended bonding modes and operations:
    Linux Channel Bonding Best Practices and Recommendations

Basics

- Virtual LAN for Z-internal connectivity, implemented in IBM Z firmware
  - ⇒ No cabling required
  - ⇒ Reliable transport
- All features of a real LAN segment supported, including VLANs.
- CHPID type IQD
- MTU sizes supported in IOCDS: 8K, 16K, 32K and 56K Recommendation is not to exceed 32K in long-running systems because of memory fragmentation
- QDIO-based interface, comparable to OSA-Express
- Device Driver: qeth
- Up to 32 HiperSockets CHPIDs with up to 4096 IP stacks each



Special Considerations

- **Synchronous transfer**: All transfers block sender till transmission completes
  - Transmission accounted to sender's CPU
  - Sender's CPU responsible for moving data to receiver's memory
  - Overloaded receivers can block senders
  - Sensitive towards receivers with insufficient CPU capacity
- No Layer 2 ↔ Layer 3 conversion
- **Promiscuous mode** as required by e.g. *Open vSwitch* available via
  - SET VNIC CHARs (recommended)
  - Bridgeport
  - Network Traffic Analyzer (NTA):
    - Requires authorization in SE per HiperSockets LAN and LPAR
    - Add'l configuration required in Linux

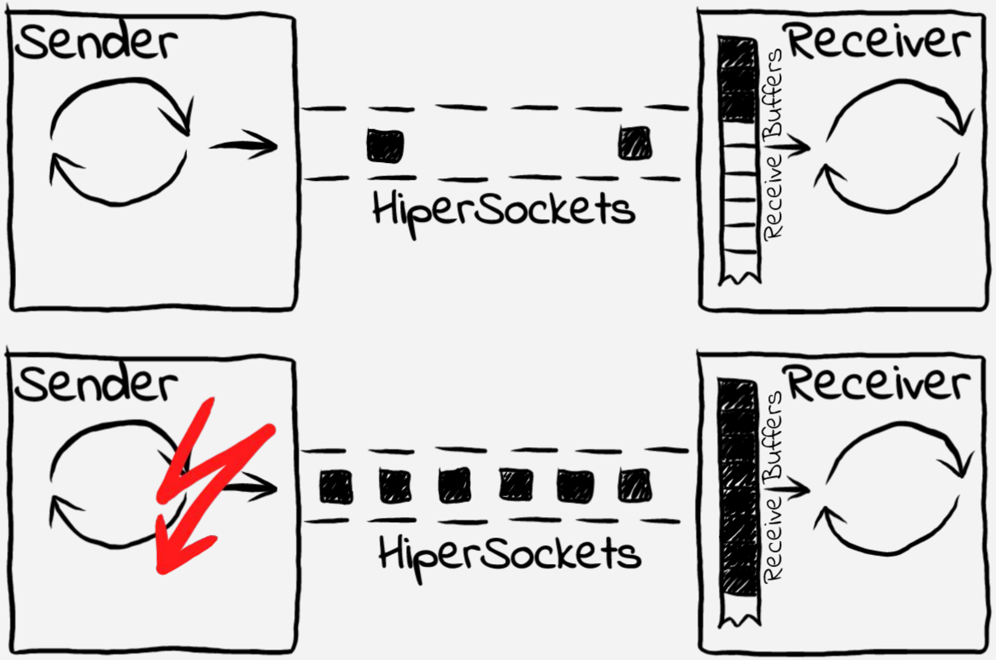


Fig 1: Synchronous HiperSockets transfers

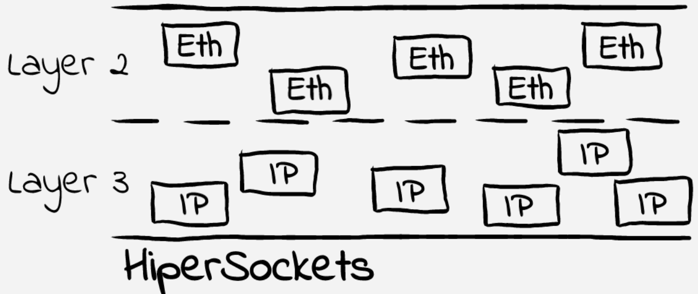


Fig 2: HiperSockets Layer2/3 separation

Summary

- **When to use**
  - Z-internal latency-sensitive workloads (i.e. request-response traffic patterns)
  - Streaming workloads, taking advantage of huge MTU sizes

- **What to consider**
  - Synchronous transfer: Sufficient CPU capacity on receiving end required
  - Streaming workloads not benefiting as much as request-response patterns
  - External connectivity requires add’l setup
  - Limit MTU size to 32k in long-running systems

- **z/OS Connectivity**
  - z/OS only supports Layer 3 for plain HS
    ⇒ Linux needs to use Layer 3, too
  - HiperSockets Converged Interface (HSCI) provides Layer 2 connectivity – but respective Linux support not available (yet)

Overview

- SMC is a ***complementary*** technology: Non-qualifying traffic uses regular transport ⇒ Optimize for regular transport, first!
- To *qualify,* traffic must be
  - within the same IP subnet
  - TCP only
  - no IPsec
- Typical complements: SMC-D with HiperSockets, SMC-R with RoCE Express only; any other regular transport (e.g. OSA) would work, too
- Applications to use `AF_SMC` instead of `AF_INET` – recompile application or use preload library via `smc_run` or

```
export LD_PRELOAD=libsmc-preload.so
```

- **Linux Distro Support:**
  - **RHEL 8**
  - **SLES 12 SP4**: Kernel level 4.12.14-95.13.1 or higher
  - **SLES 15 SP1**
  - **Ubuntu 18.10** or later

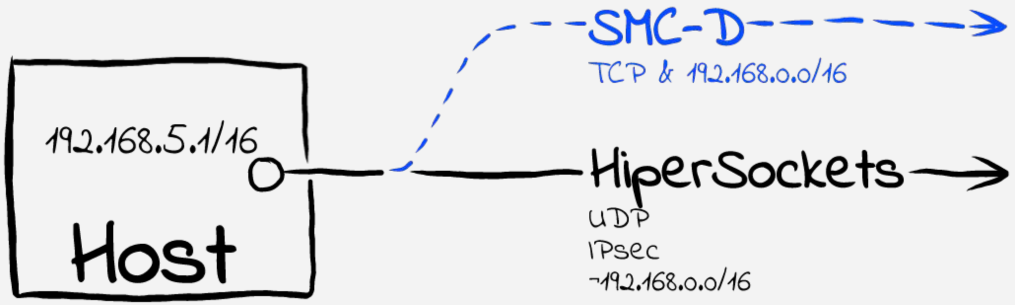


Fig 1: SMC-D sample illustration

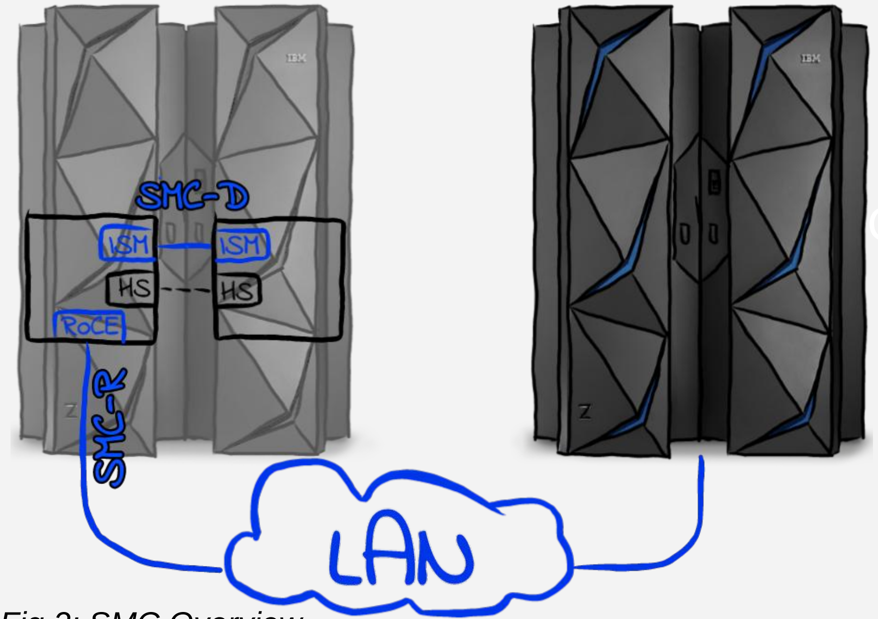
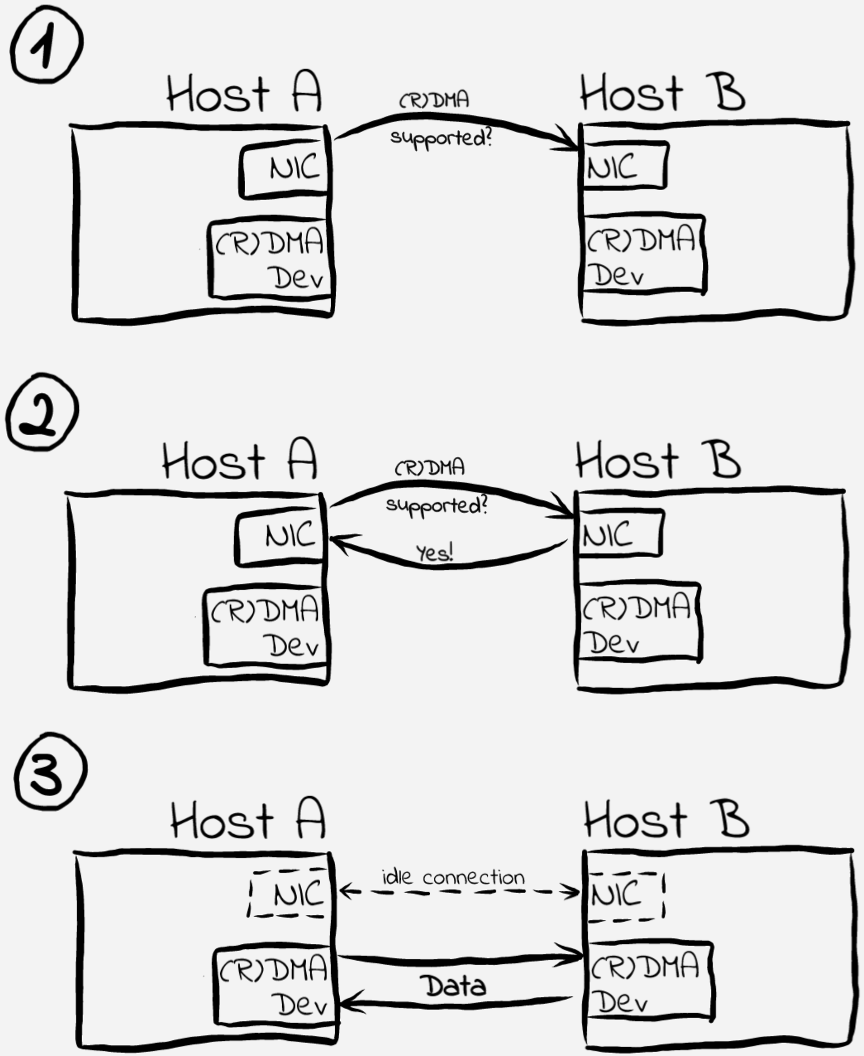


Fig 2: SMC Overview

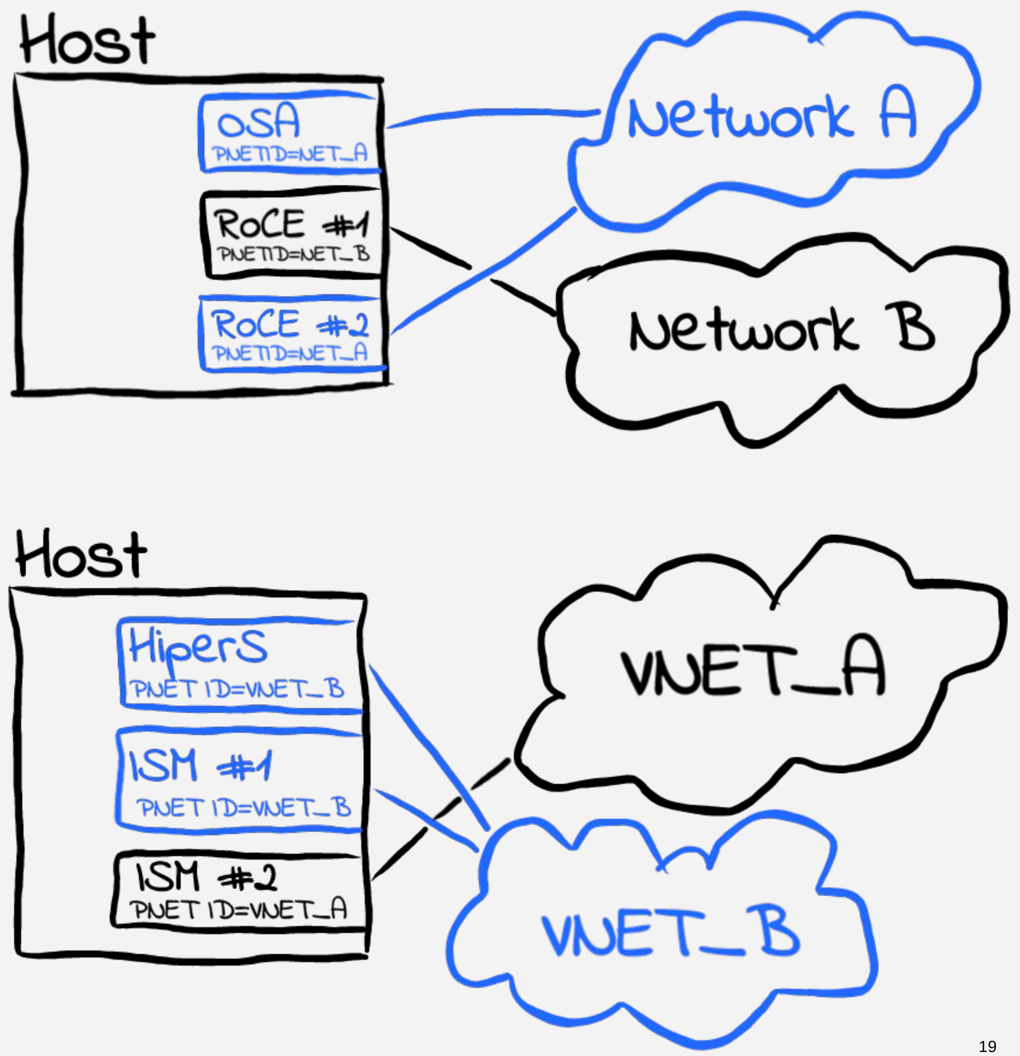
Connection Setup

- For each new TCP connection:
  - Start out with a regular TCP/IP connection, advertising (R)DMA capabilities
  - If traffic qualifies and peer confirms:
    Negotiate details about the (R)DMA capabilities & connectivity
  - Switch over to an (R)DMA device for actual traffic depending on the peers' capabilities
  - Regular TCP connection through NICs remains active but idle



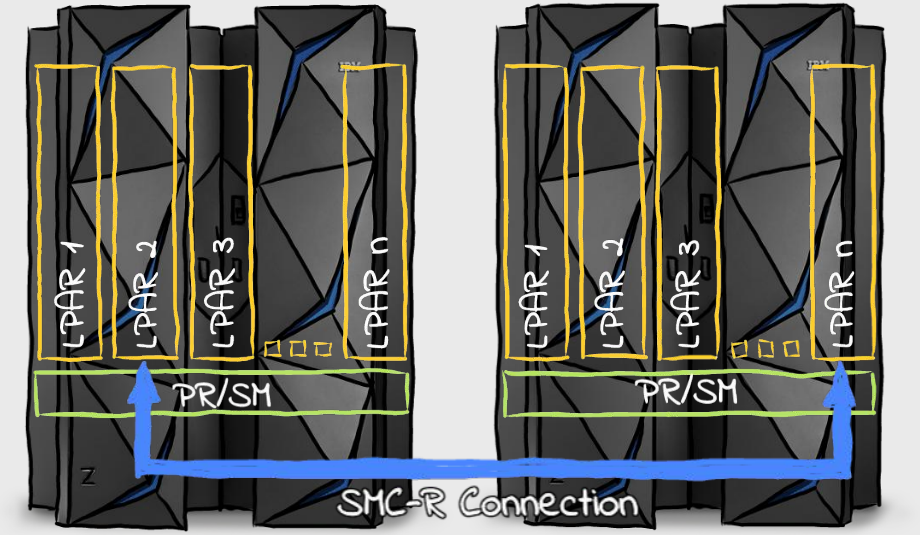
PNET IDs

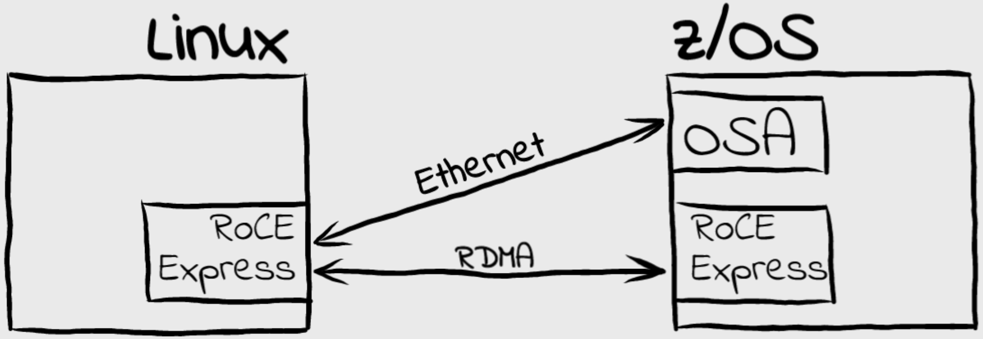
- **PNET ID**: *Physical network identifier*
- Customer-defined value to logically group NICs and RDMA adapters connected to the same physical network within a host
- Defined in
  - IOCDS for any of OSA, RoCE, HiperSockets or ISM, or
  - using `smc_pnet` tool (SMC-R only, all of the above and virtual networking facilities, e.g. z/VM vNICs)
- *Typically* associate
  - OSA and RoCE cards, or
  - HiperSockets and ISM devices
- **Note**: PNET IDs help to locate a suitable (R)DMA device for a given NIC ***within a host***. The peer can use totally different PNET IDs (as long as the correct devices are grouped)



SMC-R Overview

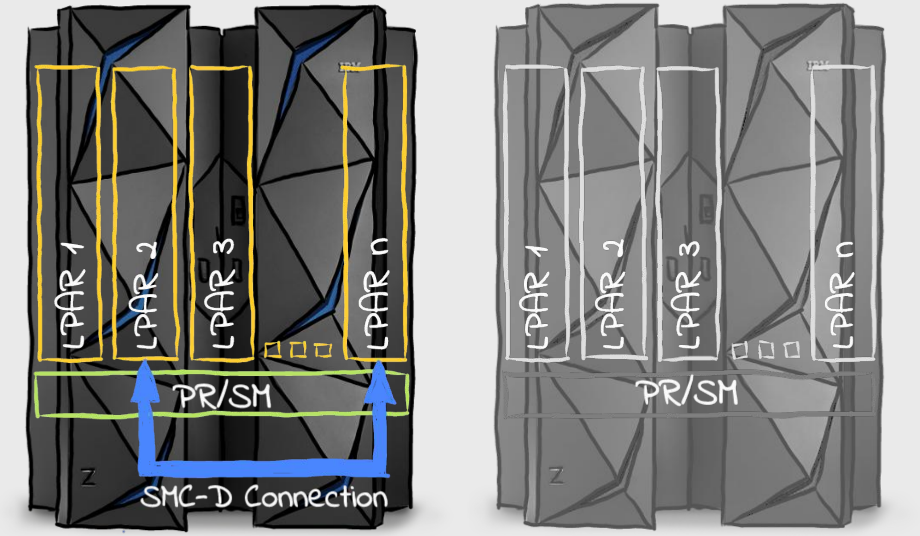
- Connectivity between Z boxes using ***RoCE Express*** cards
- IBM Z hardware requirements
  - IBM z12EC and z12BC or later
  - LinuxONE Emperor and Rockhopper or later
  - Classic and DPM mode supported
- Use OSA or RoCE card for regular connectivity
- PNET ID configuration
  - IOCDS (recommended), or
  - `smc_pnet`
- **Note**:
  - Linux on Z can use a single RoCE card for regular and RDMA traffic!
  - No link failover!





SMC-D Overview

- Z-internal connectivity using ***Internal Shared Memory*** (ISM) devices
- IBM Z hardware requirements
  - IBM z13 (requires driver level 27 (GA2)) and z13s, or later
  - LinuxONE Emperor and LinuxONE Rockhopper, or later
  - Classic mode only (i.e. DPM not supported)
- ISM devices
  - *Virtual* PCI network adapter of new VCHID type ISM
    - No PCI bus usage
    - No extra hardware required
  - 32 ISM VCHIDs per Z, 255 VFs per VCHID (8K VFs per Z total)
    I.e. the maximum no. of virtual servers that can communicate over the same ISM VCHID is 255
  - Each ISM VCHID represents a unique (isolated) internal network, each having a unique Physical Network ID
- PNET ID configuration
  - IOCDS
  - Use HiperSockets, OSA or RoCE cards for regular connectivity



Summary: SMC-R

- **When to use**
  - Low latency
  - Low CPU cost
  - High availability built into protocol (no Linux support yet)

- **What to consider**
  - Applies to a subset of overall traffic only:
    ⇒ Optimize for regular case!
  - IPsec & UDP not supported
  - Peers must be in same IP broadcast domain
  - Slightly increased memory requirements
  - Legacy applications might not benefit
  - No failover support (yet)
  - Simplify setup by using RoCE Express for RDMA and non-RDMA traffic

- **z/OS Connectivity**
  - For now, z/OS limited to use of a single RoCE device when connected to Linux

Summary: SMC-D

- **When to use**
  - Low latency
  - Low CPU cost
  - Very high throughput
  - Use all the time, e.g. to accelerate HiperSockets or shared sevices traffic

- **What to consider**
  - Applies to a subset of overall traffic only:
    => Optimize for regular case!
  - IPsec, UDP not supported
  - Peers must be in same IP broadcast domain
  - DPM mode not supported
  - Slightly increased memory requirements

- **z/OS Connectivity**
  - No limitations

Agenda

- **Part I: Common Linux on Z Networking Facilities**
- **Part II: Environment-specific Networking Facilities and Considerations**
  - **z/VM Facilities**
    - VSWITCH & Guest LAN
    - IUCV & Virtual CTC
  - **z/VM Considerations**
    - Networking Cards
    - HiperSockets
    - SMC
    - z/OS Connectivity
  - **Docker**
- **References**

VSWITCH

- Simulated network switching device
- Provides high availability and link aggregation of up to 8 OSA ports
- Supports both, Layer 2 (keyword `ETHERNET`) and Layer 3 (keyword `IP`) devices
  - Layer 2: OSA ports form LAG, providing fast fail-over and load balancing (⇒ higher throughput)
  - Layer 3: OSA ports used in fail-over mode only
- Supports LACP (IEEE 802.3ad) with *shared* OSA ports
- z/VM guests exploiting a VSWITCH require vNICs coupled to VSWITCH
- Configure vNICs just like regular OSA devices
- Guest access restricted
- **Notes**:
  - Supports OSA-Express only, no RoCE Express!
  - Supports LACP (IEEE 802.3ad) with *shared* OSA ports

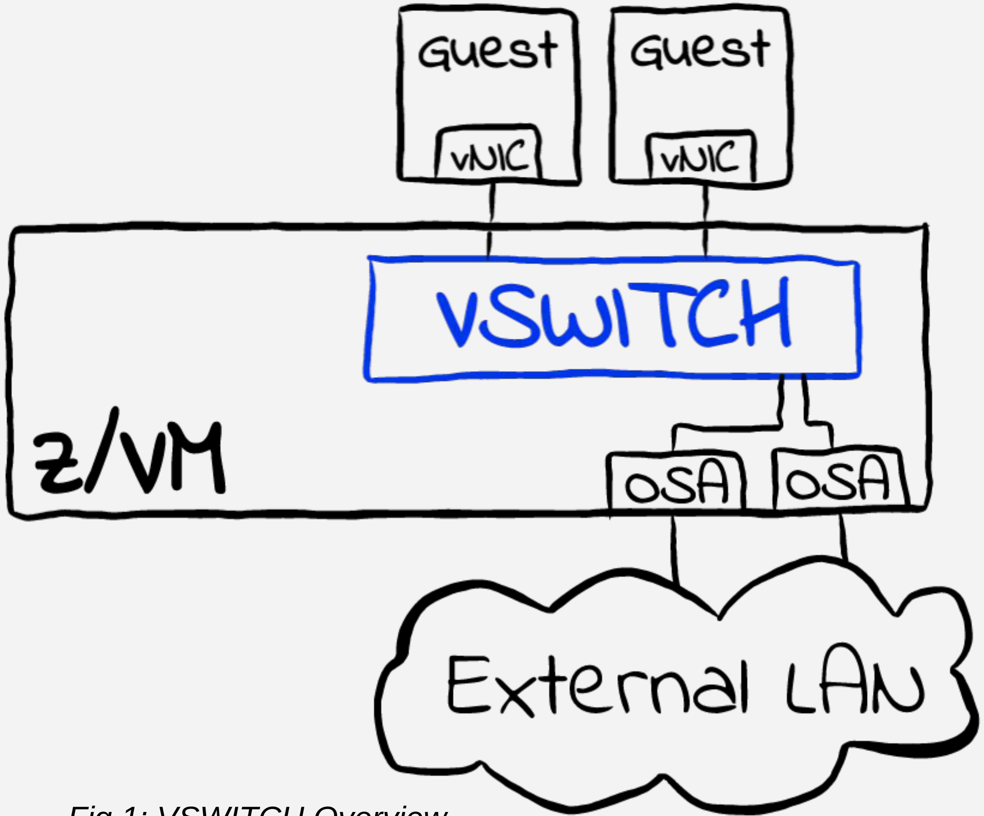


Fig 1: VSWITCH Overview

Bridgeport

- Layer 2 only: Extend existing HiperSockets to z/VM VSWITCH (or vice versa), forming a single broadcast domain
- Only one primary bridgeport at a time, multiple secondaries. If primary fails, one of the secondaries becomes the new primary
- VSWITCH-attached OSA provides external connectivity for HS without any extra routing setup required
- No obligation to attach any OSA-Express uplink ports
- Consider moving guests from VSWITCH-attached to HS-attached for higher efficiency

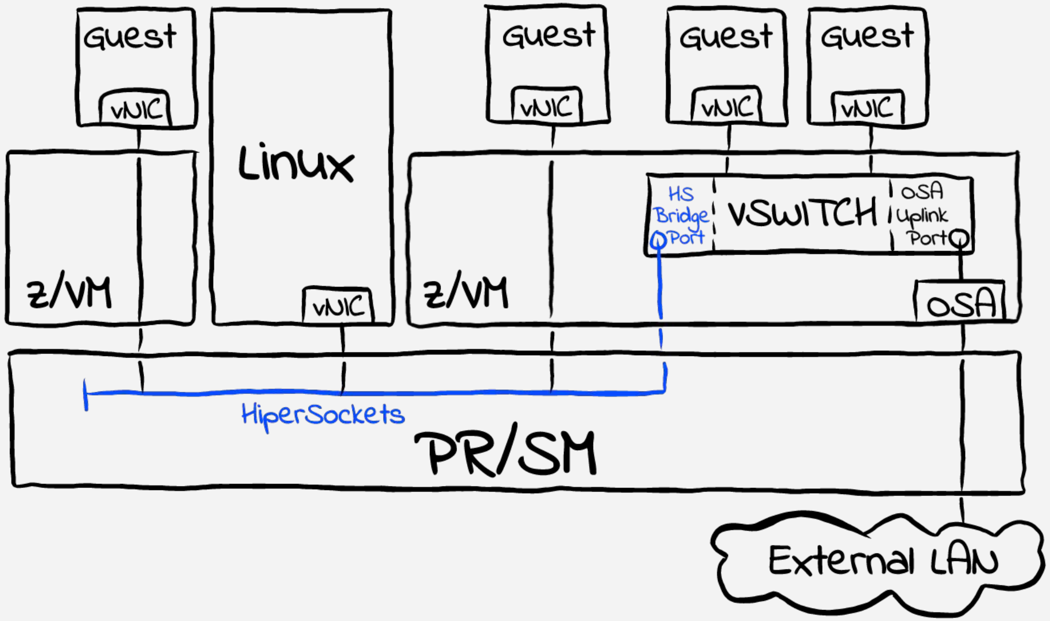


Fig 1: VSWITCH with Bridgeport Extension

Guest LAN

- Simulated LAN segment
- Either plain QDIO (Layer 2 or Layer 3) or HiperSockets (Layer 3 only – implies synchronous data transfer!)
- Guest access can be restricted
- Functional equivalence to a VSWITCH without attached OSA
- **Main purpose**: Simulate entire network topologies within z/VM prior to deployment without need of IOCDS modifications or actual cabling

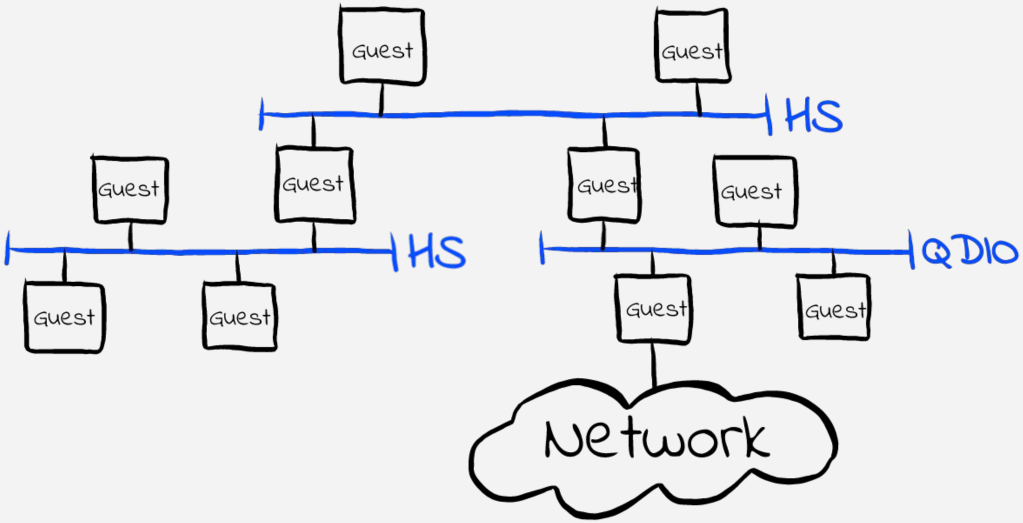


Fig 1: Sample Guest LAN-based simulated network

Summary

- **When to use**
  - **VSWITCH**
    - External connectivity
    - Simplifies link aggregation
    - Provides high availability
    - Can increase throughput
    - VSWITCH Bridgeport:
      - External connectivity for HS without routing
      - Simplicity through a single broadcast domain for HS and VSWITCH
  - **GuestLAN**
    - Simulate LAN setups
    - Offers MTU ≫ 9K with HiperSockets

- **What to consider**
  - **VSWITCH**
    - Attached z/VM guests benefit from bonding setup
    - **Bridgeport**: z/VM guests to attach to HiperSockets preferably
  - **Guest LAN**
    - No external connectivity without routing

- **z/OS Connectivity**
  - **VSWITCH Bridgeport:** HS-attached guests require QEBSM support, which is not available in z/OS

NETIUCV

- **Available on *SLES* and *Ubuntu* only**
- **Virtual point-to-point connection between two z/VM guests**
- **No bus ⇒ no eavesdropping by other guests possible**
- **Alternative: Virtual CTCA**
  - Also provides virtual point-to-point connection
  - More complicated setup
  - Performance worse than NETIUCV

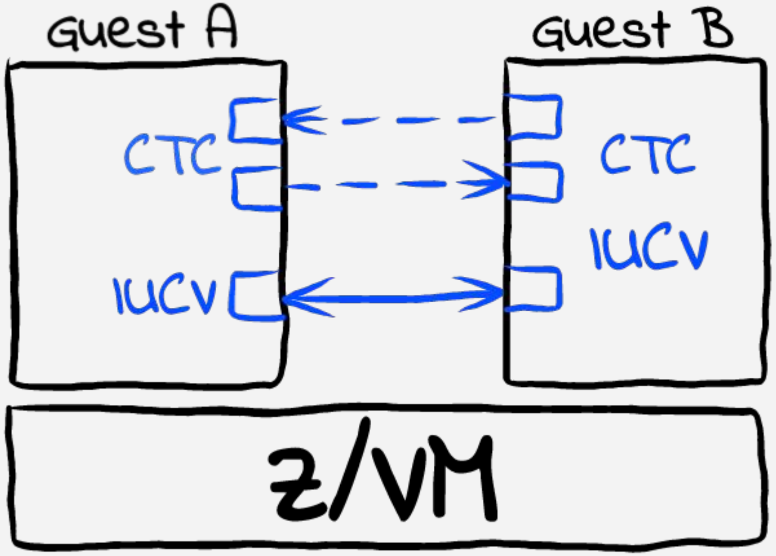


Fig 1: NETIUCV and CTC overview

```
$ modprobe netiucv
# Setup connection to guestB (peer)
$ echo guestB>/sys/bus/iucv/drivers/netiucv/connection
# Configure device
$ ip addr add 192.168.2.1/16 dev iucv0
$ ip link set up dev iucv0
$ ip addr show iucv0
6: iucv0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 9216
qdisc fq_codel state UNKNOWN group default qlen 50
    link/slip
    inet 192.168.3.1/16 scope global iucv0
       valid_lft forever preferred_lft forever
```

Fig 2: Sample NETIUCV setup

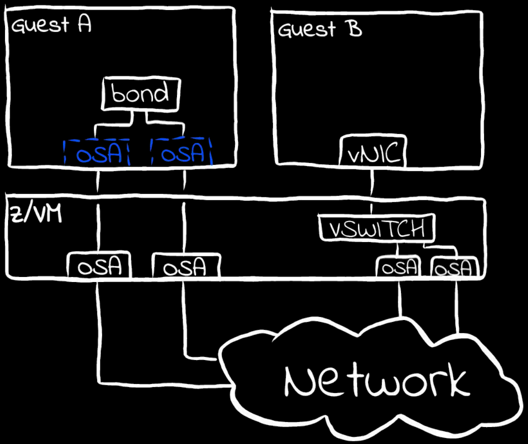
Summary

- **When to use**
  - Direct connection between two peers only
  - Protection against eavesdropping required

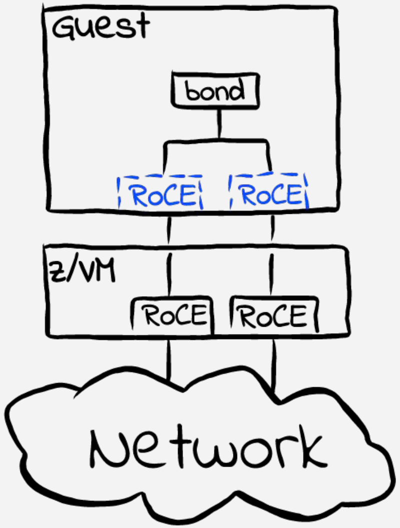
- **What to consider**
  - Simple setup
  - No interaction with z/VM admin required
  - Security aspect based on lack of 3rd parties, but no encryption involved
  - Alternative CTC requires more complicated setup, offers less performance than NETIUCV

OSA-Express

- Attach OSA device to Linux guest:
  `#CP ATTACH <devno_range> to <guest>`
- **Configuration**: Like in LPAR
- **Channel Bonding**:
  - Configure like LPAR case
  - Configuration required for *each* guest
- **Alternative**: Attach up to 8 OSA ports to VSWITCH



RoCE Express



- Attach PCI FID to Linux guest:
  `#CP ATTACH PCIFUNCTION <FID> to <guest>`
- **Configuration**: Like in LPAR
- **VSWITCH:** Not supported
- **Channel Bonding**: Configure like LPAR case

HiperSockets

- Attach OSA device to Linux guest:
  `#CP ATTACH <devno_range> to <guest>`
- **Configuration**: Like in LPAR
- **VSWITCH:** Attach as bridgeport

Summary

- **When to use**
  - Direct-attach OSA, HiperSockets and RoCE for optimum performance

- **What to consider**
  - VSWITCH offers one-stop configuration for OSA and HiperSockets via link aggregation and bridgeport respectively
  - Channel bonding through VSWITCH can share OSA ports across multiple VSWITCH instances

SMC

- IOCDS allows PNET ID assignment for NICs (OSA and RoCE), HiperSockets and ISM devices only
- I.e. vNICs as used with VSWITCH do not inherit PNET IDs from attached OSA ports
- Recommendation:
  - Direct-attach OSA, RoCE, HiperSockets and ISM devices in z/VM guests to simplify SMC usage
  - Otherwise, use *smc_pnet* to configure PNET IDs manually for vNICs (SMC-R only)

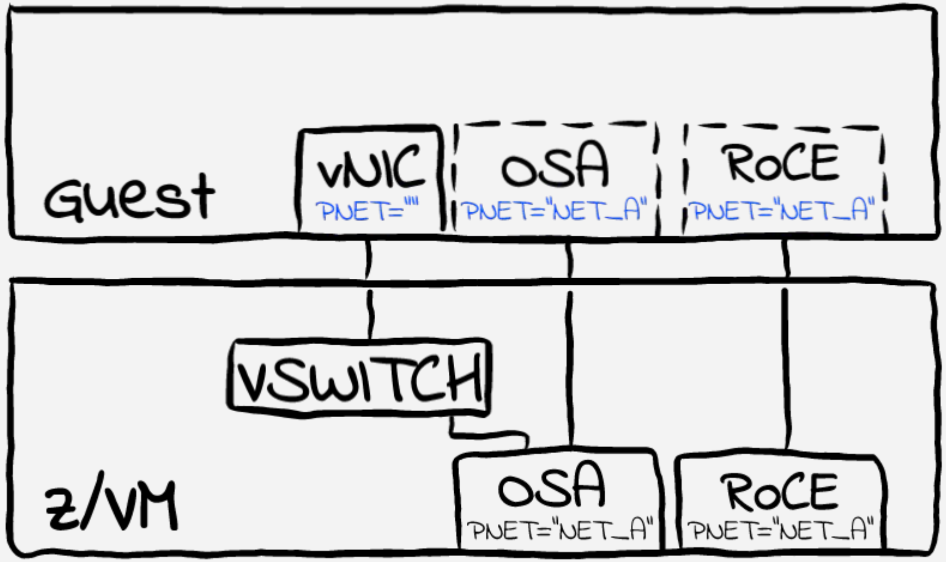


Fig 1: PNET IDs in z/VM overview

Docker Considerations

- Docker containers run in isolated environments, includes networking
  ⇒ *Prevents access to host's networking facilities*
- Various options for containers' network setup exist – defaults to bridged setup with containers in extra subnet
- To lift network isolation, use
  `docker run --network host <...>`
- **Direct-attached devices**
  - Not accessible with network isolation in place
  - (OSA, RoCE, HiperSockets): No benefit, as tap devices used by Docker hardly add any overhead
- **SMC**
  - Default setup violates SMC's same-subnet prerequisite
  - Provide container with direct access to host's IP interface by using option `--network host`
  - Modify containers to utilize `AF_SMC`
- **z/OS**:
  - ***NICs***: No limitation
  - ***HiperSockets***
    - Layer 3 only in z/OS
    - No Layer 2 ⇔ Layer 3 conversion ⇒ Layer 3 devices in host requires
    - Routing in host required ⇒ performance impact (limited, but measurable)

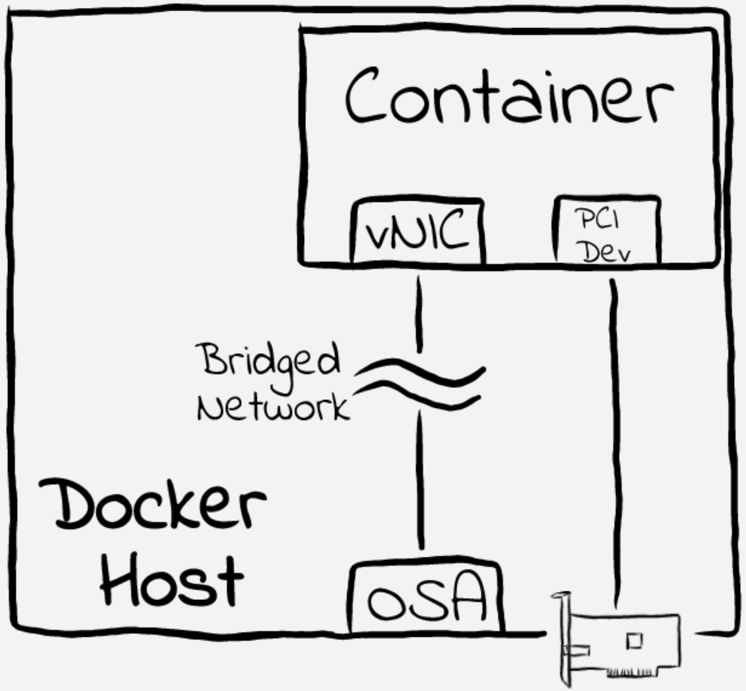


Fig 1: Docker Container with isolated network, but direct access to PCI device

References

- **Linux on Z (technical):**
  https://www.ibm.com/developerworks/linux/linux390/
- **SMC for Linux on Z:**
  https://linux-on-z.blogspot.com/p/smc-for-linux-on-ibm-z.html
- **Network Tuning Recommendations**
  https://www.ibm.com/developerworks/linux/linux390/perf/tuning_networking.html#net
- **Blogs**
  - **Linux On Z Distributions News**
    https://linuxmain.blogspot.com/
  - **Linux On Z Latest Development News**
    https://linux-on-z.blogspot.com/
  - **Containers on Z, primarily *Docker***
    https://containersonibmz.com/