Dynamically Exploit a HiperSockets Network

Dynamically exploit a HiperSockets network interface between SAP Application Server and a DB2 Database on IBM z Systems

Applies to:

Any SAP technology or product with a SAP Application Server and a DB2 database on IBM z Systems.

Summary

HiperSockets devices provide the fastest TCP/IP communication between applications running in different LPARs within a single IBM z Systems machine. Providing that the SAP Application Server (AS) under Linux for IBM z Systems and the Database system (DB) both reside on the same physical hardware (CEC), then the option to utilize the IBM z Systems internal HiperSockets network interface is available to you. This is true regardless of whether your Linux system runs under z/VM or in a native LPAR.

This article can help you to configure and dynamically activate a HiperSockets network interface. No IPL, no reboot, no SAP restart required. Your application stays continuously available.

Authors: Rainer Himmelsbach, Mark Perry, Volker Schoelles

Company: IBM Deutschland Research & Development GmbH

Created on: 7 April 2009

Updated on: 3 November 2015

Author Bio Rainer Himmelsbach

Rainer Himmelsbach graduated Dipl. Ing. in “IT & Electronics Engineering” in Berlin in 1988. He started his career at a Endress+Hausser subsidiary as a software developer.

Rainer joined IBM in 1991 and worked as team lead of the System Test test team for RMF, the z/OS® Resource Measurement Facility™. In this role he closely worked together with other IBM teams in Boeblingen, Poughkeepsie, Tucson and Endicott.

In 2007 he took the team lead position of the IBM COV-Team (Customer Oriented Validation) for SAP on IBM z Systems. He is responsible for certifications of new z/OS and Linux for IBM z Systems releases in the context of SAP and for a customer oriented validation of the Business Continuity for SAP on IBM z Systems solution.

Author Bio Mark Perry

Mark Perry is an independent contractor who has many years development and test experience in the areas of z/OS, Linux, High Availability networks and SAP. He has worked on several joint IBM/SAP projects in the Boeblingen Lab from 1999 through to 2009.

Author Bio Volker Schoelles

Volker Schoelles is a software developer at IBM Deutschland Research & Development GmbH, Boeblingen. He holds an 'Informatik Diplom' from the university of Kaiserslautern, Germany, and has worked for IBM since 1985. In 1995 he joined the IBM/SAP platform team which developed the SAP on z Systems solution. Since 2003 he has been responsible for the High Availability solution for SAP on z Systems and he is now working to extend this solution to other IBM server platforms.

Table of Contents

Introduction

HiperSockets devices provide the fastest TCP/IP communication between applications running in different LPARs within a single z Systems machine. Because it is a virtualized network, its total capacity is not limited by any physical medium (as is Ethernet), but rather by the number and speed of the processors, and the memory-bus throughput of the z System itself. Each succeeding generation of z System, by introducing more and faster processors, and higher memory-bus throughput, directly increases the total capacity of the HiperSockets.

HiperSockets are supported by these operating systems:

- z/OS
- z/VM
- Linux on IBM z Systems native LPAR
- Linux on IBM z Systems as z/VM guest
- z/VSE

Each operating system has its own TCP/IP stack configuration techniques. Here we will document what is required for both z/OS and Linux on IBM z Systems. In all cases the important goal of fully utilizing the HiperSockets network interface is primarily achieved through the increase of each stacks' TCP/IP buffers, and secondarily by increasing driver related buffers.

See Related Content to learn how Hipersockets can enhance high availability in a Business Continuity for SAP on IBM z Systems environment.

It is assumed that you will implement the recommended HiperSockets Frame Size of 16KB, which results in a TCP/IP MTU (Maximum Transmission Unit) of 8192 bytes.

3-Step Approach

This 3-step approach leads you to the fastest TCP/IP communication available for IBM z Systems

1. Define HiperSockets devices
2. Configure TCP/IP z/OS
3. Configure TCP/IP for Linux for IBM z Systems

Define the HiperSockets devices

The first step towards utilizing HiperSockets is to define them to the z Systems as an IQD type CHPID. Use z/OS components HCD and IOCP to define IQD type CHPIDs. The HiperSockets CHPID can then be shared between LPARs that need to communicate with one another. Within the LPARs any TCPIP stack can utilize the HiperSockets using a dedicated device number triplet defined on the HiperSockets CHPID. These device triplets then need to be configured online to the Operating Systems (OS's) hosting the SAP AS and DB.

For details regarding HCD, IOCP, HiperSockets in general please see chapter **Related Content**

Configure the TCP/IP stacks for z/OS

The TCP/IP buffers are specified on the TCPCONFIG statement using the options TCPRCVB and TCPSENDB for the receive and send buffers respectively. For TCP/IP buffer size the recommendation is to have at least 8 times the (largest) MTU value of the HiperSockets network interfaces, so for an MTU of 8192 bytes this means TCP/IP buffer size 65536 bytes.

The HiperSockets interface name here is HIPER plus a 2 digit hexadecimal suffix which represent the CHPID number of the associated HiperSockets. In the following examples we assume that the HiperSockets has been defined via HCP and IOCP to CHPID=E4.

z/OS 2.1 and above with VIPA support

Starting with z/OS 2.1 z/OS supports the new INTERFACE statement also for configuration of HiperSocket interfaces in an IP4 network. This is now the recommended method to define a HiperSocket interface.

Add or modify the following statements in the TCP/IP PROFILE dataset:

```
...
TCPCONFIG INT 10 SENDG TRUE UNRESTRICTL TCPRCVB 65536 TCPSENDB 65536
...
DEVICE  VLINK1   VIRTUAL  0                ; Virtual device for SOURCEVIPA
LINK    VLINK1   VIRTUAL  0 VLINK1         ; configuration
...
INTERFACE   HIPERE4                       ; HiperSocket name
   DEFINE   IPAQIDIO                      ; HiperSocket interface type
   IPADDR   10.101.4.5/26                 ; V4 ADDRESS + SUBNET MASK
   SOURCEVIPAINTERFACE VLINK1             ; Associate SOURCEVIPA to VLINK1
   CHPID    E4                            ; HiperSocket CHPID
...
HOME
10.101.4.210            VLINK1            ; Static IP for SOURCEVIPA
...
START HIPERE4
...
```

Then restart TCP/IP.

z/OS 2.1 and above without VIPA

Starting with z/OS 2.1 z/OS supports the new INTERFACE statement also for configuration of HiperSocket interfaces in an IP4 network. This is now the recommended method to define a HiperSocket interface.

Add or modify the following statements in the TCP/IP PROFILE dataset:

```
...
TCPCONFIG INT 10 SENDG TRUE UNRESTRICTL TCPRCVB 65536 TCPSENDB 65536
...
INTERFACE   HIPERE4                       ; HiperSocket name
   DEFINE   IPAQIDIO                      ; HiperSocket interface type
```

```
   IPADDR   10.101.4.5/26                  ; V4 ADDRESS + SUBNET MASK
   CHPID    E4                             ; HiperSocket CHPID
...
START HIPERE4
...
```

Then restart TCP/IP.

z/OS 1.13

Add or modify the following statements in the TCP/IP PROFILE dataset:

```
...

TCPCONFIG INT 10 SENDG TRUE UNRESTRICTL TCPRCVB 65536 TCPSENDB 65536
...
DEVICE IUTIQDE4 MPCIPA
LINK   HIPERE4  IPAQIDIO IUTIQDE4
...
HOME
...
10.101.4.2           HIPERE4
...
BEGINROUTES
...
ROUTE 10.101.4.0/26 = HIPERE4 MTU 8192
...
ENDROUTES
...
START IUTIQDE4
...
```

Then restart TCP/IP.

Configure the TCP/IP stacks for Linux for IBM z Systems

There are two major Distributions of Linux for IBM z Systems. One is RHEL from RedHat and the other one is SLES from Novell/Suse. Current levels of each are RHEL 7.1 and SLES 12.

This section explains how to configure TCP/IP stacks for RHEL 7.1 and SLES 11. The SLES 11 configuration can be used with SLES 12 as well.

If your Linux runs under z/VM then follow these recommendations:

- To simplify the administration of multiple Linux Guests the HiperSockets device triplet RDEVs should be mapped to VDEVs that are the same for all Guests. (Note that the device triplet can be dynamically attached to a running Guest.)
- Define sufficient storage to each Guest, so that any extra TCP/IP buffers are not an issue.
- Provide sufficient CP's and CP SHARE to each Guest to maximize HiperSockets performance.

Note: HiperSockets performance is closely related to the number and speed of the processors.

RHEL 7.1

Modify or create configuration file `/etc/sysconfig/network-scripts/ifcfg-hsi0` for the device triplet xxxx,yyyy,zzzz

```
# IBM QETH
DEVICE=hsi0
NETTYPE=qeth
TYPE=Ethernet
ONBOOT=yes
PORTNAME=OSAPORT
SUBCHANNELS=0.0.xxxx,0.0.yyyy,0.0.zzzz
BOOTPROTO="static"
IPADDR="10.101.4.14"
NETMASK="255.255.255.192"
NETWORK="10.101.4.192"
OPTIONS="buffer_count=128"
```

If the device triplet is not yet attached to the z/VM Linux Guest do that now.

Bring the new interface online:

```
ifup hsi0
```

SLES 11 / SLES 12

Add a new permanent UDEV rule to ensure the interface name remains constant across reboots.

Modify or add to file: `/etc/udev/rules.d/70-persistent-net.rules :`

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="qeth", KERNELS=="0.0.xxxx",
ATTR{type}=="1", KERNEL=="hsi*", NAME="hsi0"
```

Modify or create a network configuration file for the new HSI on dev xxxx:

`/etc/sysconfig/network/ifcfg-hsi0`

```
BOOTPROTO='static'
BROADCAST=''
ETHTOOL_OPTIONS=''
IPADDR='10.101.4.7/26'
MTU=''
NAME='Hipersocket (0.0.xxxx)'
NETWORK=''
REMOTE_IPADDR=''
STARTMODE='auto'
USERCONTROL='no'
```

If the device triplet is not yet attached to the z/VM Linux Guest do that now.

Bring the new hardware online to Linux:

```
hwup hsi0
```

Configure the qeth device triplet xxxx,yyyy,zzzz:

```
qeth_configure -o buffer_count=128 0.0.xxxx 0.0.yyyy 0.0.zzzz 1
```

Note: the above command does not issue error messages, only return codes. It should create a new udev rule for hsi0 in `/etc/udev/rules.d`.

Bring the new interface online:

```
ifup hsi0
```

Switch between HiperSockets and Gigabit Ethernet

Let's assume your actual network communication goes over Gigabit Ethernet. This section explains how you switch dynamically between HiperSockets interface communication and Gigabit Ethernet communication. This can be useful for tests or for maintenance reasons.

On Linux the /etc/hosts file includes routing information for TCP/IP. The information maps IP address to host name.

Here is an example of a /etc/hosts file:

```
……
# IP address set to connect to host via Gigabit Ethernet
# 10.100.1.1      <host name A>
# 10.100.1.2      <host name B>
# 10.100.1.3      <host name C>
……
# IP address set to connect to host via HiperSockets
10.101.1.1<host name A>
10.101.1.2<host name B>
10.101.1.3<host name C>
……
```

For test and validation reasons you may want to keep the Gigabit Ethernet entries. To activate communication over HiperSockets simply add a comment to the Gigabit Ethernet definitions like in the sample. Save the file and then only the HiperSockets interface will be used.

Note: The information in /etc/hosts is mostly used at boot time and when a new process connects to a host. Consider a reboot if you must pickup the most recent changes in /etc/hosts.

Related Content

IBM Redbook “HiperSockets Implementation Guide”
http://www.redbooks.ibm.com/abstracts/sg246816.html?Open

Business Continuity for SAP on IBM z Systems
http://publibfp.dhe.ibm.com/epubs/pdf/iapacs07.pdf

z/OS 2.1 HCD manuals
http://www.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.cbd/cbd.htm

IBM Redbook “IBM z Systems Connectivity Handbook” SG24-5444 (refer to Chap.9)
http://www.redbooks.ibm.com/redpieces/abstracts/sg245444.html?Open

z/OS 2.1 Communications Server manuals (includes TCP/IP)
http://www.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.cs3/cs3.htm

Copyright