# *IBM Technical Brief*

## SAP® Customer Relationship Management 7.0 on IBM Z®

**Authors:**

**Seewah Chan**
**Paul DeLessio**
**Dr. Paul Lekkas**

**Document Owner:**

**Paul D. DeLessio**
**SAP on IBM Z Performance**
**IBM Poughkeepsie**

**Version: 1.00**
**Date: March 1, 2019**

**Filename: IBM_Z_SAP_CRM70.pdf**

# Table of Contents

# TABLES

# FIGURES

## Disclaimers

Neither this document nor any part of it may be copied or reproduced in any form or by any means or translated into another language, without the prior consent of the IBM Corporation. IBM makes no warranties or representations with respect to the content hereof and specifically disclaims any implied warranties of merchantability or fitness of any particular purpose. IBM assumes no responsibility for any errors that may appear in this document. The information contained in this document is subject to change without any notice. IBM reserves the right to make any such changes without obligation to notify any person of such revision or changes. IBM makes no commitment to keep the information contained herein up to date.

Performance data and customer experiences for IBM and non-IBM products and services contained in this document were derived under specific operating and environmental conditions. The actual results obtained by any party implementing any such products or services will depend on a large number of factors specific to such party's operating environment and may vary significantly. IBM makes no representation that these results can be expected or obtained in any implementation of any such products or services.

The information in this document is provided "as-is" without any warranty, either expressed or implied. IBM expressly disclaims any warranties of merchantability, fitness for a particular purpose or infringement.

## Trademarks

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

AIX, Db2, DS8800, FICON, FICON Express, IBM, IBM logo, OSA, OSA Express, HiperSockets, System Storage, IBM Z, zEC12, z13, z14, z/Architecture, z/OS, z/VM, and zSeries, are trademarks or registered trademarks of the International Business Machines Corporation in the United States and other countries.

SAP, the SAP logo, and all other SAP product and service names mentioned herein are trademarks or registered trademarks of SAP SE in Germany and in several other countries all over the world.

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

SUSE Linux Enterprise Server and SLES are registered trademarks of Novell, Inc., in the United States and other countries.

Other products may be trademarks or registered trademarks of their respective companies. Information is provided "AS IS" without warranty of any kind. Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, 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.

Other company, product, or service names may be trademarks or service marks of others.

## Feedback

Please send comments or suggestions for changes to delessio@us.ibm.com.

# 1 Introduction

SAP Customer Relationship Management (CRM) is part of the SAP Business Suite [1] and can be rapidly deployed to manage and improve customer satisfaction and relationships. CRM is widely used by many midsize and large organizations in all industries and sectors. It is utilized to complement a business's core needs by managing and storing customer information, such as accounts, appointments, marketing email, contact info, leads, sales opportunities, and much more, in a central location for use with analytics to help build stronger customer relationships.

SAP CRM is typically run on a satellite system that is integrated with an Enterprise Resource Planning (ERP) or other external systems. CRM activities include a wide spectrum of queries ranging from simple to complex data access for report generation and some background data exchanges between the CRM and other external systems.

This paper describes testing that was performed to simulate the high load data exchange activity from an external system into an SAP CRM system running on IBM Z. The test activity was similar to what would occur from an SAP ERP or other external system importing data into SAP CRM. The focus was to demonstrate SAP CRM running on IBM Z with a Db2 database server on z/OS and SAP application servers on Linux in a co-located z14 environment. The test experiments reaffirmed IBM Z co-location benefits and some best practice findings for Linux on Z application servers.

This CRM high load data exchange test did not use any of the SAP standard benchmarks. The test workload used shared similar behaviors with SAP ERP workloads. Within the test effort, there was a look at common platform independent settings and a couple of Linux on Z application server performance specific options.

The SAP on IBM Z Performance Team, that performed this study, is in Poughkeepsie NY. The team's mission is to examine SAP performance on IBM Z systems. SAP SE and IBM Z have a long partnership that provides a robust portfolio and significant value to businesses. SAP using a Db2 for z/OS server with Linux on Z clients provide a highly available co-located solution with efficient total cost of IT ownership. For many enterprises, SAP business software on IBM Z plays a key role in critical areas, from business planning to customer relationship management.

# 2 CRM Test Workload

There are typically two types of background data exchange activities performed on a CRM system:

1. Import, or replication of data, from an external system into CRM with subsequent actioning on the data.
2. Export, which includes processing of data in CRM (e.g. “billing run”, “calculating loyalty points”, “marketing campaign execution”, etc.) and then replication to an external system (e.g. “external mail provider sending out bills”, “emails on customer loyalty points accounts”, etc.).

Test efforts focused on background data exchange type 1, not 2. The first type often is a high load data exchange into CRM. In this case, the master data or transactional data is created or changed on an external system and the update is then propagated to CRM. What differentiates this workload from a typical CRM mass import data exchange is that there is no sending system. Thus, the test had to bypass the middleware actions which include receiving and converting the data and the associated database and application server activities. What was then done, as it would be for any CRM mass import, is the processing of a converted document. The SAP test program creates the converted document and triggers the data import into the CRM database as part of the normal CRM mass data exchange processing.

Business Transactions, such as appointments, sales orders, service orders, opportunities, leads, and many more, can be created using the same underlying SAP CRM order transaction process. There is little difference in the process of creating an appointment, activity, service ticket, marketing email, contract or sales order transaction with the exception being that some transactions are heavier than others.

## 2.1 Workload Description

The actual workload for this effort was built from a sample program that comes with SAP CRM and is used by SAP developers and others to simulate data exchange activity for a functional check into CRM. The program was called with various parameter values with each unique call being labeled a “variant”. To circumvent the limitations of the sample program and to broaden its execution scope to a mass load, like that of a CRM background mass data exchange, the test program was called many times successively with multiple copies running in parallel. To be more precise, the program was called with 32 unique variants in parallel and each of these variants were called many times in a stream. Figure 1 below illustrates the 32 parallel streams of the variant calls.

| | **Variant Stream 1** | **Variant Stream 2** | **Variant Stream 3** | ***Variant Stream …*** | **Variant Stream 30** | **Variant Stream 31** | **Variant Stream 32** |
|---|---|---|---|---|---|---|---|
| **Call 1** | Variant 1 | Variant 2 | Variant 3 | *Variant …* | Variant 30 | Variant 31 | Variant 32 |
| **Call 2** | Variant 1 | Variant 2 | Variant 3 | *Variant …* | Variant 30 | Variant 31 | Variant 32 |
| **Call 3** | Variant 1 | Variant 2 | Variant 3 | *Variant …* | Variant 30 | Variant 31 | Variant 32 |
| **Call 4** | Variant 1 | Variant 2 | Variant 3 | *Variant …* | Variant 30 | Variant 31 | Variant 32 |
| ***Call …*** | **…** | **…** | **…** | **…** | **…** | **…** | **…** |

**Figure 1: Parallel Stream Variant Calls**

To better simulate an actively used CRM system, over 1.8 million Business Partner records were pre-loaded into the CRM database prior to the test measurements. The main distinction between this workload and an actual CRM data exchange were slightly lower processing demands on the application and database servers by omitting a few middleware layers. The test objects were “Appointment” objects which were chosen over others to minimize additional setup required by the other object types. Functionally, the data exchange processing is the same for all CRM object types.

## 2.2 Platform Independent Options

There are a few performance related options that are fundamental to a typical SAP system regardless of its underlying operating system or hardware platform. They are common to many different SAP workloads and are not specific to SAP CRM. Though the settings are made on the application server, they have performance impact on both the application and database servers. The following lists the platform independent options that affect workload performance and are applicable to this SAP CRM study:

- “Packet size” - number of objects processed per variant call.
- SAP number range buffering.
- SAP update process configuration.

The platform independent options described here for SAP CRM are also common with many other SAP workloads including SAP ERP. The goal of exploring these options is to achieve successful CRM mass data exchange measurements with minimal performance bottlenecks. The derived settings for these options were rolled into the base for all measurements described in this document. There are some tradeoffs with various settings which are discussed in the next sections.

### 2.2.1 Packet Size

The “packet size” is the number of objects to process per variant call. From an application program viewpoint, it’s basically the processing unit size. With a small packet size, the application program will incur more variant calls to process a given number of objects within a batch stream. With each variant call there exists a fixed amount of processing overhead, such as program initialization. Thus, a small packet size may have higher processing overheads due to more variant calls. Conversely, with a large packet size, the program would incur fewer variant calls, thus, lower processing overheads. However, the application program may consume more memory and acquire more semaphores per variant call with a large packet size. The increased accumulation of memory and semaphores with a large packet size can increase the contention of each and counteract the reduced processing overheads from fewer variant calls. Adjusting packet size is an option to balance between these costs.

To get to an optimal number, measurements with various packet sizes were compared by elapsed time per object. Packet size comparison was done by normalizing the batch elapsed time to the number of processed objects. Results of the effort showed that using the default packet size of 1000 gave the test workload reasonable performance with no obvious bottleneck or inhibitor.

Modifying the "packet size" is not unique to the CRM data exchange workload. As observed in other SAP workloads, e.g. SAP Banking Account Settlement, the number of objects, bank accounts in that case, per processing unit can be modified.

The packet size adjustment is not a one size fits all workloads. For this CRM data exchange study, SAP recommended a default of 1000 for the packet size and testing showed little reason to differ from that.

### 2.2.2 Number Range Buffering

Some applications may have requirements to assign unique numbers to certain processed objects. For instance, a Sales and Distribution (SD) application may need to assign unique sales order or invoice numbers. Similar to the other applications, the document number has to be unique for the SAP CRM mass data exchange workload. However, there is performance overhead associated with assigning numbers especially when database calls are invoked for each assignment.

The Number Range Buffer is an SAP feature to boost performance when assigning numbers. Instead of fetching the number individually from the database, a single access to the buffer, residing locally on the application server, is more expedient. The buffer is replenished from the database only when it is depleted. Thus, the performance overhead associated with number assignment can be drastically reduced. The Number Range Buffer is used by many SAP applications, including the SD application in SAP ERP as well as this SAP CRM mass data exchange workload.

The default for the SAP CRM document object is 10 per application number range request. Comparisons between various number range buffer settings were made for this CRM data exchange evaluation. Increasing the number range buffer size to 100 showed some noticeable performance benefit of elapsed time reduction. However, there were negligible difference when going beyond this. Therefore, all subsequent measurements in this effort used a number range buffer size of 100.

There is one caution, using the number range buffer may introduce gaps in the number assignments. Increasing the size increases the chance of number range gaps but has the benefit of reducing number range buffer requests and the associated overhead. If the gaps are not critical to an application, it may be an adjustment worthwhile to consider.

### 2.2.3 Update Process Configuration

The SAP Update Process is used to mitigate the effect of time-consuming changes made to the database. Database update activity can affect application server performance by causing dialog and batch front-end process queueing. SAP can minimize front-end queueing on the application servers by performing updates asynchronously. With asynchronous updates, dialog and batch process queueing is reduced as update activity is offloaded from the mainline processing path. This offload of update activity from mainline processing may improve workload performance. Like what is found in SAP ERP workloads, this study found that the CRM high load data exchange had asynchronous update activity.

SAP asynchronous update performance is further improved through common setup considerations. Considerations such as *partitioning asynchronous update protocol tables* and *a few parameter settings* which include directing update requests to be processed to the local server and instance that made the request. These are common performance tuning options that many SAP applications may exploit.

There are (3) frequently accessed SAP **asynchronous update protocol tables** (VBHDR, VBMOD, VBDATA). Update processing can be improved by partitioning them. Partitioning these tables may spread the database inserts to multiple partitions thus improving the concurrency for the insert activities.

**A few parameter settings**, in the SAP profile configuration files, were looked at in the tests. These were options that worked for this effort and are not a specific rule. Aside from making sure there were enough update threads configured, below are some parameters related to update processing that were set:

| | |
|---|---|
| dynp/luw_id_format | = 2 |
| rdisp/vb_dispatching | = 0 |
| rdisp/vb_key_comp | = SYNR/WPNR/HOST/DATE/TIME/STMP |
| rdisp/vbname | = $(rdisp/myname) |

Parameter “**dynp/luw_id_format=2**” says to use the “**rdisp/vb_key_comp**” parameter for guidance to partition update processing tables “VBDATA, VBMOD and VBHDR” (VBLOG).

Parameter “**rdisp/vb_key_comp=SYNR/WPNR/HOST/DATE/TIME/STMP**” is the partitioning key order of consideration for VBLOG tables. For this effort, these tables were simply partitioned by SYNR or instance number, to spread the work across (8) SAP instances.

Parameter “**rdisp/vb_dispatching=0**” says to send update requests to the server shown in profile parameter “**rdisp/vbname**”. If parameter “**rdisp/vbname**” is set to “**$(rdisp/myname)**”, update requests will be processed by the server and instance that made the request. This reduces cross-system processing and makes more efficient use of application server resources by keeping heavy asynchronous update activity local to the server and instance that made the request.

# 3 System Environment

The SAP test system consisted of a database server and three application servers all co-located in a single IBM z14 3906-M01. The database server resided in a dedicated LPAR running z/OS 2.2 and Db2 12 for z/OS. The application servers resided in three dedicated LPARs running SUSE Linux Enterprise Server (SLES) 12 SP3. The z14 had 33 cores and 2TB processor memory of which 31 cores and a subset of the memory as listed below were used for this effort.

## 3.1 Hardware Environment

**IBM Z Database Server:** The SAP database server was in a dedicated LPAR with 4 processors and 512 GB of real memory.

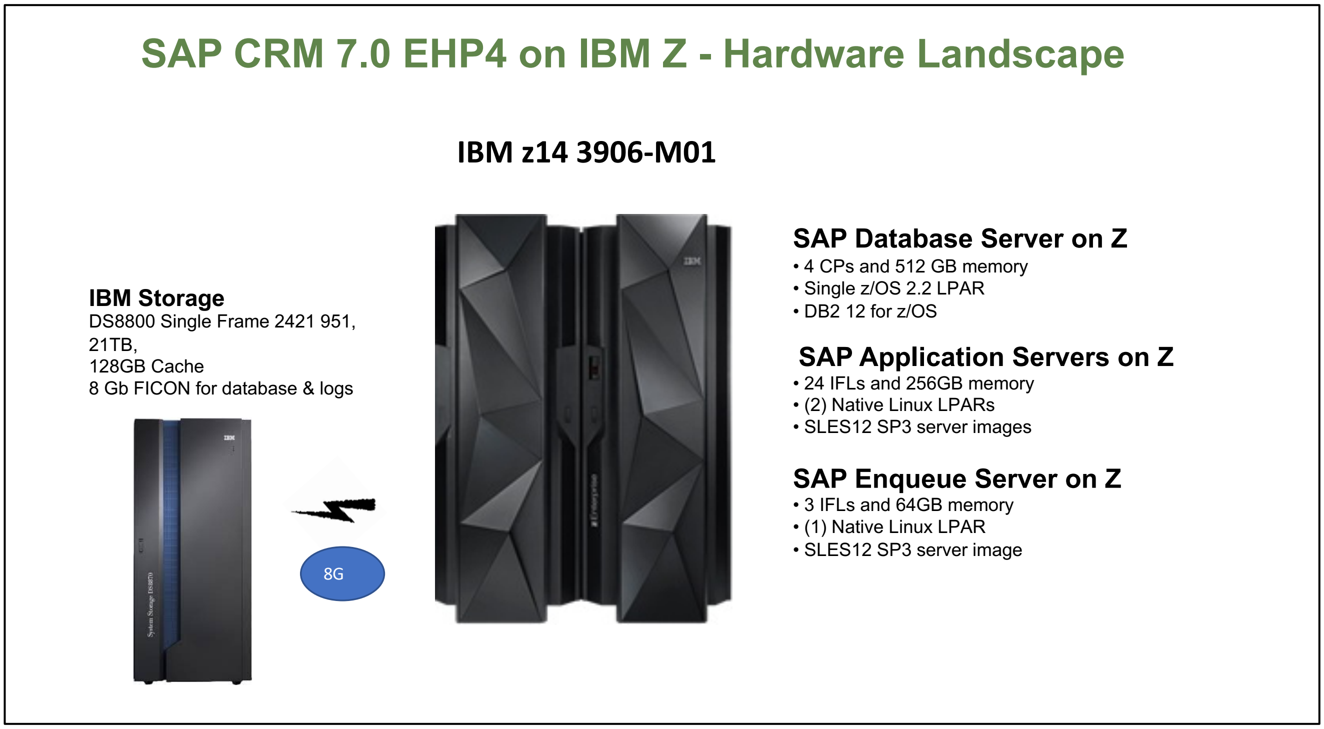
**Storage:** A dedicated IBM System Storage Server DS8800 Model 2421-951 was used for the database, logs and backups. The capacity of the unit was 21 TB 15K rpm HDD drives and 128 GB cache memory. The storage server was attached to the z14 via FICON Express8S LX connections.

The active database used (58) 3390 Mod 54 volumes or about 3TB of allocated space.

**Application Servers:** The SAP application servers were in three dedicated LPARs, two dialog application server images plus one standalone enqueue server image consisting of the following:

- (2) Dialog application servers with (12) IFLs and 128 GB of memory each.
- (1) Message Enqueue server image with (3) IFLs and 64 GB of memory.

**Network:** HiperSockets connectivity was used for all network connections to exploit one of the IBM Z co-location performance features. HiperSockets connectivity offers high speed in-memory communication between LPARs and reduces network latency. There is also the advantage of eliminating adapter cards and physical cabling needs which reduces potential physical failure points and associated costs.



SAP CRM 7.0 EHP4 on IBM Z - Hardware Landscape

IBM z14 3906-M01

**IBM Storage**
DS8800 Single Frame 2421 951,
21TB,
128GB Cache
8 Gb FICON for database & logs

**SAP Database Server on Z**

- 4 CPs and 512 GB memory
- Single z/OS 2.2 LPAR
- DB2 12 for z/OS

**SAP Application Servers on Z**

- 24 IFLs and 256GB memory
- (2) Native Linux LPARs
- SLES12 SP3 server images

**SAP Enqueue Server on Z**

- 3 IFLs and 64GB memory
- (1) Native Linux LPAR
- SLES12 SP3 server image

**Figure 2: Conceptual View of SAP CRM 7.0 EHP4 on IBM Z - Hardware Landscape**

## 3.2 Software Environment

**z/OS**
z/OS release 2.2

**Db2 for z/OS**
Db2 12

**Db2 Connect**
IBM Db2 Connect - Version 11.1 FP0 special 35734.

**Linux**
SLES 12 SP3

**SAP Product & Component Version**

| Product | Release | SP Stack | Short Description of Product Version |
|---|---|---|---|
| SAP NETWEAVER | 7.5 | 07 (04/2017 | SAP NETWEAVER 7.5 |
| SAP CRM | 7.0 | 18 (06/2016) | SAP CRM 7.0 / NW7.01 |
| EHP4 FOR SAP CRM 7.0 | EHP4 FOR SAP CRM 7.0 | 06 (05/2017) FP | EHP4 FOR SAP CRM 7.0 |

**Table 1: SAP Product Versions**

| Software Component | Release | Level | Highest Support Package | Short Description of Software Component |
|---|---|---|---|---|
| SAP_BASIS | 750 | 0007 | SAPK-75007INSAPBASIS | SAP Basis Component |
| SAP_ABA | 750 | 0007 | SAPK-75007INSAPABA | Cross-Application Component |
| SAP_GWFND | 750 | 0007 | SAPK-75007INSAPGWFND | SAP Gateway Foundation |
| SAP_UI | 750 | 0007 | SAPK-75007INSAPUI | User Interface Technology |
| ST-PI | 740 | 0006 | SAPK-74006INSTPI | SAP Solution Tools Plug-In |
| SAP_BW | 750 | 0007 | SAPK-75007INSAPBW | SAP Business Warehouse |
| MDG_FND | 749 | 0007 | SAPK-74907INMDGFND | MDG Foundation |
| SAP_AP | 750 | 0004 | SAPK-75004INSAPAP | SAP Application Platform |
| SAP_BS_FND | 748 | 0006 | SAPK-74806INSAPBSFND | SAP Business Suite Foundation |
| WEBCUIF | 748 | 0006 | SAPK-74806INWEBCUIF | SAP Web UI Framework |
| BBPCRM | 714 | 0006 | SAPK-71406INBBPCRM | SAP CRM |
| CRM_PLUS | 714 | 0006 | SAPK-71406INCRMPLUS | SAP CRM PLUS |
| FINBASIS | 748 | 0006 | SAPK-74806INFINBASIS | Financial Basis |
| FSCM_CCD | 618 | 0006 | SAPK-61806INFSCMCCD | Financial Supply Chain Management: Customer Dispute Processing |

**Table 2: SAP Component Versions**

# 4 Measurements

## 4.1 Measurement Methodology

This CRM on IBM Z study focused on a couple of application server related settings that affect Linux on Z performance. Specifically, focus was on the settings for vertical CPU polarization and receive packet steering. Other IBM Z platform specific recommended setup, like the use of HiperSockets, dedicated processors etc. were baked into all measurements, as were platform independent option values, previously described in section 2.2.

In a typical customer IT landscape, the CRM system is often integrated with other systems, such as an ERP system. There could be a large amount of data exchanges between the CRM and these other systems. Though the data exchanges are bidirectional, this study only focused on the data flow into the CRM system. The imported data could be pushed into the CRM system from an external system via dialog processes. Alternatively, the imported data could be pulled into the CRM system via batch processes. In this study, the pull model was used by the CRM data exchange measurements.

Prior to each measurement, the CRM database was restored to an identical starting point to ensure measurement consistency. This starting point included over 1.8 million pre-loaded Business Transaction records to simulate an actively used CRM system verses an empty system that would rarely exist. For each measurement, the following performance indicators were captured to gauge the system resource requirements and to provide performance insights:

- The average elapsed time of batch processing.
- CPU utilizations of the database and application servers.
- External Throughput Rate (ETR) in units of “million objects per hour” which is calculated by dividing the total number of objected inserted by the average batch elapsed time.
- Internal Throughput Rate (ITR) is the ETR normalized to 100% processor utilization.

The most important performance metric of this study is the ETR, being the rate of data imported into the CRM system. ITR is then derived from the ETR and CPU utilization as a reliable basis for measuring processor capacity [2].

## 4.2 IBM Z Platform Specific Options for SAP Application Server

SAP applications in general are more resource intensive on the application servers. The SAP CRM background data exchange was no exception. Therefore, the performance focus of this study was on application server options.

Following the general performance tuning guidelines for the SAP Application server on IBM Z, the baseline test scenario was configured:

- **HiperSockets** were used for the SAP database, application, and Enqueue server connectivity. HiperSockets exploit processor memory communication without any physical networking connections across servers within the same physical machine. They provide a low-latency network for better performance.
- **Dedicated native LPAR**s were used for all server images.

- 12 **IFL**s with **SMT** enabled were used for each of the two application servers. SMT allows more than one active instruction stream per core and may boost core processing capacity to improve performance.
- The **standalone SAP enqueue** server was isolated in its own dedicated LPAR with 3 IFLs.

There exist many Linux configuration and parameter settings that could influence SAP application server performance. This was not an exhaustive evaluation of all possible Linux configuration options. This study explored two of the more promising performance influencing parameters based on other studies, namely **vertical CPU polarization** and **receive packet steering** (RPS).

### 4.2.1 Vertical CPU Polarization

By default, Linux on Z sets dispatching mode to horizontal CPU polarization, where all CPUs are dispatched evenly for about the same amount of time by the hypervisor. When the dispatching mode is switched to vertical CPU polarization, certain CPUs are dispatched for a longer time than others. This may preserve cache contents and minimize cache misses and disruptions. Thus, vertical CPU polarization may increase processing efficiency and provide potential performance benefits.

The following diagram shows an example of the polarization state before and after the switch from horizontal to vertical CPU polarization.

```
To display CPU Polarization (default) setting:

# lscpu -e

       CPU NODE DRAWER BOOK SOCKET CORE L1d:L1i:L2d:L2i ONLINE CONFIGURED POLARIZATION
       ADDRESS
       0   0    0      0    0      0    0:0:0:0         yes    yes        horizontal   0
       1   0    0      0    0      0    1:1:1:1         yes    yes        horizontal   1
       2   0    0      0    0      1    2:2:2:2         yes    yes        horizontal   2
       3   0    0      0    0      1    3:3:3:3         yes    yes        horizontal   3

       etc.

To enable vertical CPU Polarization
# chcpu -p vertical
       Successfully set vertical dispatching mode


To verify Vertical CPU Polarization setting enablement:
# lscpu -e

       CPU NODE DRAWER BOOK SOCKET CORE L1d:L1i:L2d:L2i ONLINE CONFIGURED POLARIZATION
       ADDRESS
       0   0    0      0    0      0    0:0:0:0         yes    yes        vert-high    0
       1   0    0      0    0      0    1:1:1:1         yes    yes        vert-high    1
       2   0    0      0    0      1    2:2:2:2         yes    yes        vert-high    2
       3   0    0      0    0      1    3:3:3:3         yes    yes        vert-high    3

etc.
```

**Figure 3: Vertical CPU Polarization setup**

For Linux in a shared LPAR, there will be a combination of vertical high, medium and low CPUs assigned dependent on the LPAR weight with respect to the resources available and the definitions of other LPARs in the CEC. For Linux in a dedicated LPAR, all CPUs are designated as vertical high when Vertical CPU Polarization is set.

### 4.2.2 Receive Packet Steering (RPS)

With advancements in network technology, the host system has become more and more of the limiting factor to achieving maximum network performance. As the growth in CPU speed tapered off, computer systems have shifted to designs with more CPU cores. In order to keep up with the increasing network load, the system must be able to distribute the work across multiple CPU cores.

Receive Packet Steering (RPS) is one method to distribute the work to multiple CPU cores and boosts performance through efficiencies in how it distributes parallel processed work. RPS creates a hash from the IP addresses and port numbers, and then uses the hash to determine which CPU to enqueue a packet of work. The hash also ensures that data packets for the same stream of data are sent to the same CPU, which helps to increase performance.

In this study, the test network interface used is a single HiperSockets interface with a single receive queue. To enable RPS for specific CPUs, to process data packets for the receive queue of the network interface, the value of the CPU position bit in the bitmap needs to be set to 1.

The following example illustrates before and after enabling RPS setting for 12 CPU cores for a HiperSockets interface named hsi2.

```
To display RPS (default) setting:
# cat /sys/class/net/hsi2/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000

To enable the RPS setting for 12 CPU cores:
# echo "ffffff" > /sys/class/net/hsi2/queues/rx-0/rps_cpus

To verify RPS setting enabled for 12 CPU cores:
# cat /sys/class/net/hsi2/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,00000000,00000000,00ffffff
```

**Figure 4: Receive Packet Steering (RPS) setup**

## 4.3 Measurement Results and Analysis

Throughout all test scenarios in this study, the configuration always had 4 general purpose CPUs for the database server, 24 IFLs for the two SAP application servers, and 3 IFLs for the SAP message enqueue server. The configuration used a total of 31 CPU cores which fit within a 3906-M01 model. It was modest in size as compared to a high-end z14 3906-M05 with 170 CPU cores.

For the two exploratory test scenarios, the independent variables were the “Vertical CPU Polarization” and the “Receive Packet Steering” settings respectively. All other setup and values were held consistent across test scenarios. These included the database/application server processor, memory, storage and network configurations; quantity of imported objects; and number of concurrent batch jobs. Prior to each test scenario, the database was restored to the exact same starting point to ensure measurement consistency.

The test objective is not meant to be a rigorous performance benchmark aiming for the optimal measurement results. Rather, it is a stress test to emulate the peak background CRM data exchange activities which a customer may experience. Apart from the normal background data exchange activities running throughout the day, the peak data exchange activities, sometimes referred to as “mass updates”, are the accumulated bulk data processing executed during certain non-peak periods. In this study, the amount of imported data and the number of batch jobs were chosen to drive the database and application servers to a sufficient load and duration (e.g. >40% CPU utilization for 30 minutes) yet with enough spare capacity left to allow the processing of other potential work.

As mentioned previously, the key performance metric of this study is the External Throughput Rate (ETR) which is calculated by dividing the total number of objects imported by the average batch elapsed time. All test scenarios in the study imported the same 1.28 million (M) objects. The calculated ETR would be to divide 1.28M objects by the average elapsed time (in hours) of the 32 concurrent batch jobs obtained from the measurement.

For the baseline scenario, the ETR achieved was 2.45M objects per hour. The CPU utilization of the database and application servers were 42% and 62% respectively. There was still plenty of processor capacity available and no observable system constraint limiting a higher ETR (e.g. running more concurrent batch jobs). However, the primary test objective of this study was not the optimal ETR. The achieved ETR of 2.45 million objects per hour demonstrated that a moderately sized IBM Z machine can support a high data import rate with the SAP CRM database and application servers co-located together. The actual import rate will vary depending on the size and complexity of the data structures for a specific customer.

| SAP CRM 7.0 on IBM Z | | | |
|---|---|---|---|
| | **Baseline** | **Vertical-high CPU Polarization** | **RPS - All CPs** |
| **Run ID** | S80510T2 | S80512T1 | S80514T1 |
| **Processor (DB server)** | z14-704 | z14-704 | z14-704 |
| **DB server O/S** | z/OS 2.2 | z/OS 2.2 | z/OS 2.2 |
| **Db2 version** | 12 | 12 | 12 |
| **Processor (App servers)** | 24 IFL | 24 IFL | 24 IFL |
| **SMT enabled** | yes | yes | yes |
| **App server O/S** | SLES 12 SP3 | SLES 12 SP3 | SLES 12 SP3 |
| **# of jobs** | 32 jobs | 32 jobs | 32 jobs |
| **Total # of objects (1.28M)** | 1,280,000 | 1,280,000 | 1,280,000 |
| **z/OS utilization** | 42% | 42% | 42% |
| **App server utilization** | 62% | 63% | 64% |
| **Avg batch elapsed time (sec)** | 1,881 | 1,877 | 1,822 |
| **ETR (million objects per hour)** | **2.45** | **2.45** | **2.53** |
| **ITR (DB server)** | 5.83 | 5.85 | 6.02 |
| **ITR (App server)** | 3.95 | 3.90 | 3.95 |

**Table 3: CRM 7.0 Measurement Summary**

For the vertical CPU polarization test scenario, the ETR at 2.45M objects per hour was equivalent to the baseline. This was likely that the system hypervisor with the baseline default of horizontal CPU polarization was already dispatching work efficiently for the CRM workload under evaluation. Also, note that the vertical CPU polarization setting is relevant only to native Linux, and not applicable to Linux guests running on other virtualization layers. (e.g. KVM, z/VM)

For the Receive Packet Steering (RPS) scenario, the ETR at 2.53M objects per hour increased by 3% as compared to the baseline. This improvement was in the low range of the benefits as seen in other studies. Nevertheless, RPS is helpful to SAP CRM data exchange. In this study RPS was set for all CPs, due to ease of implementation, though other settings potentially may offer higher improvement. Also, note that RPS is not limited to native Linux but is applicable to Linux guests running on other virtualization layers as well.

# 5 Summary

CRM helps companies improve customer relationships. It can be integrated into a company's IT landscape for the purpose of attracting and retaining customers by facilitating better customer interactions and satisfaction.  Data is propagated into CRM systems often with a high load data exchange.  This study demonstrated a high load data exchange into SAP CRM on IBM Z using a co-located Z landscape that benefitted from the use of the Z HiperSockets performance feature, which offers high speed in-memory communications and reduces network latency.

Data propagation into CRM systems drives significantly heavier application server activity than database server.  Update activity found during this CRM high load data exchange may be offloaded from mainline processing and executed asynchronously by SAP.  These workload characteristics are similar to what are often found in other SAP systems such as SAP ERP.  Because of these common workload characteristics, tuning of update activity, packet size, number range buffering and other performance options are similarly tuned in a CRM high load data exchange as they would be tuned in other SAP systems.

The typical heavy application server utilization of this CRM workload lead to a focus of the study centered on the Linux on Z application servers.  There are many possible options to adjust within Linux, a number of which were looked at in another SAP on IBM Z Performance Team study using SAP core banking on IBM Z [3].  Two Linux tuning parameters explored in that study, Vertical CPU Polarization and Receive Packet Steering, were explored in this CRM high load data exchange effort.  These parameter settings demonstrated similar benefit to application server performance, in both studies.

IBM Z is the choice of most of the world's largest financial institutions and a high percentage of large government entities.  It is a secure, trusted, logical and cost-effective choice for financial transactions globally with billions of business transactions daily.  All of these government and commercial entities benefit from better managed customer relationships.  SAP CRM would be a good complement to the core business of these large IBM Z customer installations.

# 6 References

[1] IBM Corp. 2016. - SAP on IBM Z Reference Architectures SAP Business Suite
https://www.sap.com/documents/2016/05/0e376aba-737c-0010-82c7-eda71af511fa.html

[2] IBM Corp 2018. - Large Systems Performance Reference for IBM Z
https://www-304.ibm.com/servers/resourcelink/lib03060.nsf/pages/lsprindex/$file/SC28118721.pdf

[3] IBM Corp 2018. - IBM Performance Report:
Performance Evaluation of SAP Application Servers on Z
http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102759