IBM Performance Report
Performance Evaluation of SAP Application Servers on Z

Authors:
Seewah Chan
Dr. Paul Lekkas
Veng Ly

Document Owner:

Veng Ly
SAP on IBM Z Performance
Version: 1.02
Date: July 16, 2018
Filename: SAP_on_IBM_Z_AppServer_Evaluation.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.

1.0 Introduction

IBM Z is designed and built to serve the most demanding enterprise applications. It offers the most robust, secure and scalable infrastructure. It is known for its zero-downtime design for business continuity, by eliminating any possible single point of failure with redundancy on both hardware and software. It excels with security features that are built into the hardware, firmware and operating systems.

With more processor cores and simultaneous multithreading option for specialty processors in the recent generations of IBM Z [1], the total processing capacity of a single physical IBM Z "box", aka Central Electronic Complex (CEC), has been drastically increased. The dramatic boost in single CEC processing capacity has provided an appealing option for customers to simplify/merge their server configurations and landscape by means of server consolidation and colocation.

Consolidating multiple servers onto a single IBM Z CEC allows exploitation of IBM internal communication (HiperSockets) without any physical network connections across many servers within the same physical machine. Thus, processing at the physical network layer is avoided. This provides a low-latency network communication and translates to better performance and throughput in addition to cost savings of minimizing physical network components like adapters, cables, switches, power and floor space.

Colocation offers better economics and simplification, and has enticed more and more customers, especially large banks, to choose this option to run their key workloads, such as the SAP core banking applications. Because of the increased demand for simplified and colocated configurations, there arose a need for a comprehensive performance study to provide best practice guidance.

To satisfy the customer colocation interests, the IBM SAP on Z Performance Team, located in Poughkeepsie, NY, conducted an extensive set of experiments to evaluate the SAP application server performance on IBM Z with the SAP Banking Account Settlement workload [2]. The key performance metric of the study is the batch elapsed time to address the critical batch window concerns which many customers face due to the operational and regulatory requirements. To optimize the batch elapsed time, various configurations and tuning options of the Linux application servers were tested and measured.

This paper documents these tests and findings. The measurements that were done were stress tests, not certified benchmarks.

2.0 Test Workload

2.1 Test Workload Characteristics

The test workload is part of the SAP core banking. It simulates a typical mass account balancing during the night processing of a retail bank. During account balancing, the system determines account balances, calculates interest and charges, etc. In general, this workload is quite resource intensive. It has to meet a critical time window within which all the processing must be completed. For example, some customers need to complete all the account balancing activity at month end within several hours in one night.

The workload is SAP Banking Services (SBS) 7.0 and has 60 million banking accounts in the database. The measurement goal is to shorten the elapsed time of the 60 million account processing.

2.2 Test Methodology

The study focused on the application server with various configurations that may influence its performance. They included OSA and HiperSockets (HS), z13 and z14, simultaneous multithreading (SMT) disabled and enabled, topology patch applied to z/VM Linux guests, native Linux LPARs, SLES12 SP1 and SP3, size of application server configuration at 64 and 128 Integrated Facility for Linux (IFLs), and Linux tuning parameters such as vertical CPU polarization, real-time scheduling policy, and receive packet steering.

The database server configuration was kept constant. For example, the number of batch jobs was held constant at 128, which is in the reasonable range for a large batch load. The database server was co-located with the application servers on the same CEC. The database server configuration was kept at 16 dedicated general processors and 1TB memory. Also, the database IO layout was kept the same.

2.3 Test Key Performance Indicators (KPIs)

To gauge the system resource requirements and to provide performance insights, the following Key Performance Indicators (KPIs) were captured for every measurement in the test scenarios:

- Elapsed time of the batch processing
- CPU Utilization on Database Server and Application Server(s)
- External Throughput Rate (ETR)
- Internal Throughput Rate (ITR)

The most important metric is the elapsed time, addressing the critical batch processing window. The External Throughput Rate (ETR) is the transaction rate, which has 60 million accounts processed by 128 batch jobs over the elapsed time. The Internal Throughput Rate (ITR) is the ETR normalized to 100% processor utilization. ITR provides a reliable basis for measuring processor capacity [3].

3.0 Test Environment

3.1 Hardware Environment

System z Database Server
The SAP database server operated within an LPAR on either an IBM z13 or z14. The LPAR was configured with 16 dedicated CPs and 1TB memory.

Database Storage
The database had a total of 60 million banking accounts. It resided on 4 IBM System Storage DS8870 subsystems and its logs resided on a dedicated IBM System Storage DS8870.

SAP Application Servers
The SAP application servers were on one of the two machines: z13 or z14.

3.2 Software Environment

z/OS
z/OS release 2.2

z/VM
6.4

Db2 for z/OS
Db2 12

Db2 Connect
IBM Data Server Driver for CLI that is shipped as part of Db2 Connect 11.01.0000

AIX
7.1

LINUX
SLES12 SP1 & SP3

SAP Banking Services 7.0
SAP Kernel 7.21 level 500

3.3 z13 Test Environment

The following figure is a graphical representation of the test environment with z13. The database server and the application servers are co-located in the same z13 CEC. In this test environment, the tests were conducted using OSA and HiperSockets connections, and the SAP application servers were configured with 64 IFLs, running on Linux as z/VM guests.

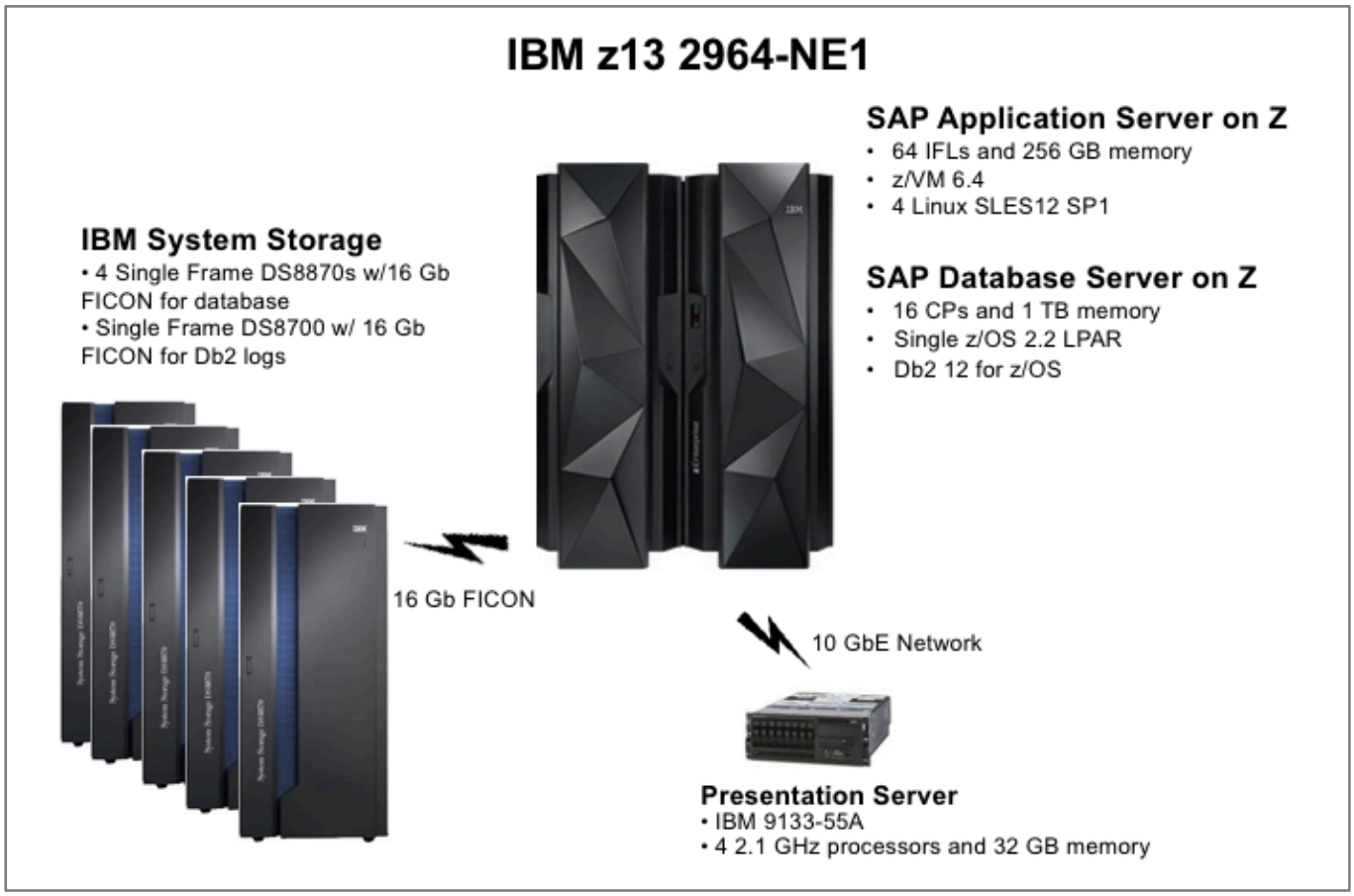

Figure 1: z13 Test Environment

3.4 z14 Test Environment

The following is a graphical representation of the test environment using z14. The test environment was similar to the z13 test environment. Most parts were kept constant. In this test environment HiperSockets was mainly used. The SAP application servers were configured to run on Linux as either z/VM guests or native Linux, with 64 or 128 IFLs in various scenarios.

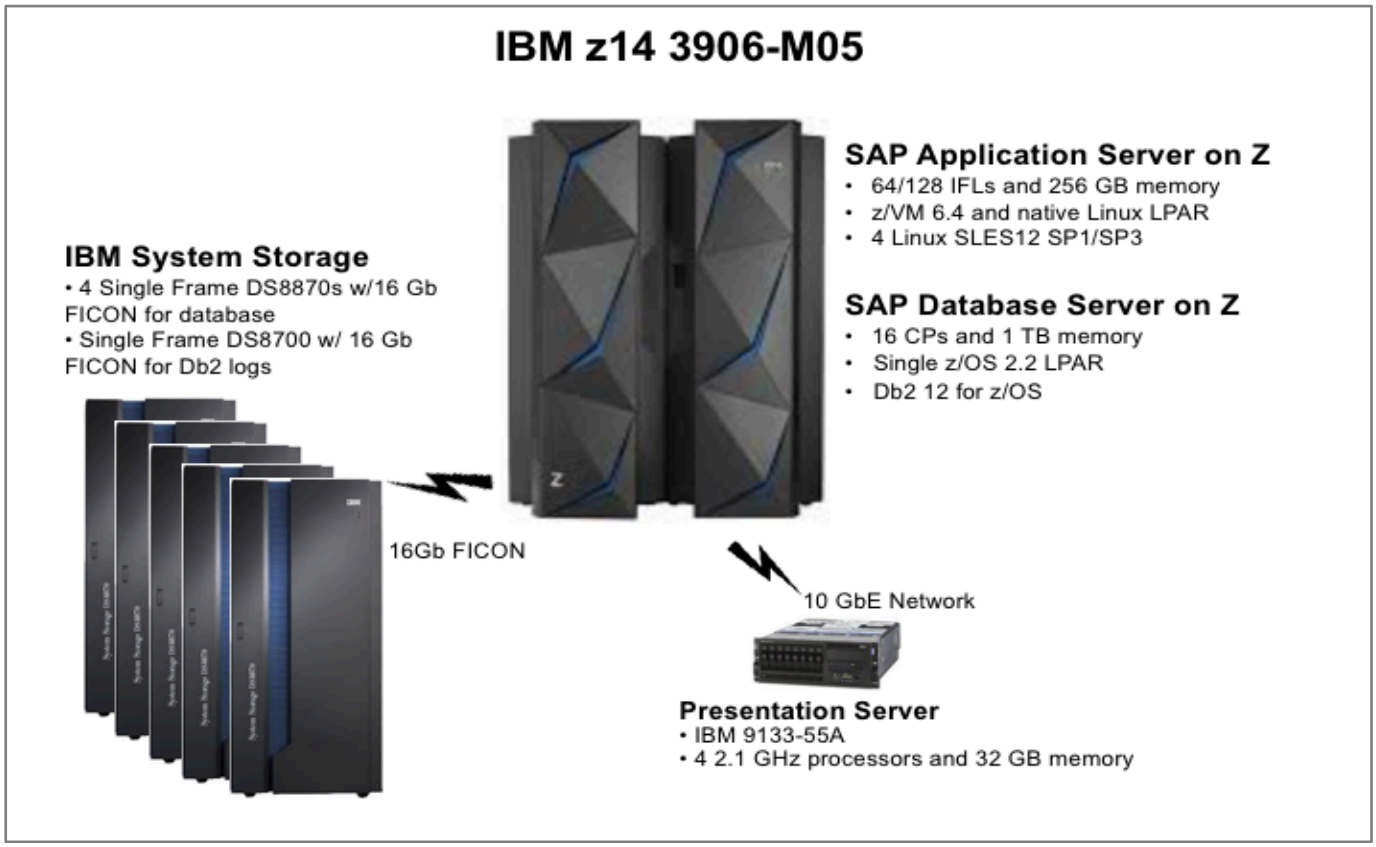

Figure 2: z14 Test Environment

4.0 Measurement Overviews

The study focused on the colocation of the SAP application servers on Linux and the SAP database server on IBM Z with the following configurations that may influence its performance:

- HiperSockets vs OSA
- z14 vs z13
- SMT disabled and enabled
- z/VM Linux guests, with and without topology patch
- Native Linux LPARs vs z/VM guests
- Linux tuning parameters (vertical CPU polarization, real-time scheduling policy, receive packet steering)
- SLES12 SP1, SLES12 SP3
- Size of application server (128 vs 64 IFLs)

The rationale of the measurement scenarios is as follows:

Network Connectivity (HiperSockets vs OSA)

Many of the SAP on Z customers use external application servers connected via OSA to the database server on Z. Colocation of multiple servers onto a single IBM Z CEC allows exploitation of IBM internal communication (HiperSockets) without any physical network cards, across servers within the same physical machine. Thus, processing at the network physical layer is avoided. This provides a low-latency network communication and translates to better performance and throughput.

Application Server Hardware Performance (z14 vs z13)

The two most recent Z processor generations are z13 and z14. IBM z14 has faster and more processor cores than z13. The study sought to highlight the performance and capacity advantages of z14.

Simultaneous Multithreading (SMT)

SMT allows two active instruction streams per core, each dynamically sharing the core's execution resources. The SMT feature is available in IBM Z processors for workloads running on the Integrated Facility for Linux (IFL). SMT can be exploited to gain potential performance benefits.

The performance benefits of SMT can vary depending on the characteristics of the workload. Highly computation-intensive batch workloads that require total core resource for each thread might not see significant performance benefits with SMT. The capacity and throughput gain per core depends on the overlap and interference between two threads. An overlap can result from many core resources being replicated so that each thread can make progress, or while one thread waits for cache miss, other threads can continue to run. Interference can be from some serialization points within the core. It can be threads sharing the same caches, thus cache misses can increase, or cause contention [4].

Virtualization Layer (Native Linux vs z/VM Linux guests)

The SAP application servers can run either on Linux as native LPARs or as guests of z/VM. z/VM is the premier virtualization layer for server consolidation. It provides benefits of ease of management, administration and deployment. Virtualization on any platform has an extra processing layer which may pose additional performance considerations. The study sought to compare performance for various sizes and numbers of z/VM systems and guests per z/VM system. The study also evaluated the performance advantage of the native Linux LPARs.

Linux Server Options (Distro Releases and Tuning Parameters)

There are concerns about new releases of Linux distros which may incur performance regressions. The tests concentrated on SLES releases 12 SP1 and SP3. The study investigated any performance difference. Linux tuning parameters, such as vertical CPU polarization, real-time scheduling policy and receive packet steering, can offer potential performance improvements which need to be measured and confirmed.

Size of the Linux Application Server (128 IFLs vs 64 IFLs)

How does the performance scale as the processing resource is increased? As the Linux application server configuration size increases, can the critical batch elapsed time be reduced? In this study, the Linux application server size was doubled from 64 to 128 IFLs. The batch elapsed time was greatly reduced.

IBM z14 has up to 170 cores. The Linux server configuration size can be increased to more than 128 IFLs. In addition, the total processor capacity can be further increased by enabling SMT. However, the number of batch jobs and a few configuration parameters needed to be adjusted accordingly to take full advantage of the additional processor threads and capacity.

5.0 Measurement Results and Analysis

5.1 HiperSockets vs. OSA on z13

The initial test scenario focus was network connectivity. Many of the SAP on Z customers use external application servers connected via OSA to the database server on Z. Thus, the baseline test configuration was using OSA connections on z13. Then the network connectivity was changed to HiperSockets (HS) per the IBM Z server consolidation recommendation.

The advantages of using HiperSockets are lower network latency and higher network throughput because it is implemented in memory. This also eliminates the need for any physical cablings for TCPIP communication within the CEC. The network traffic data are driven by system processors, not delegated or off-loaded to OSA card. This may result in higher CPU cost.

The SAP application server configuration had 4 SLES12 SP1 Linux servers as z/VM guests running on a z13 with 64 Integrated Facility for Linux (IFLs). Two z/VM LPARs were configured. Each z/VM had 32 IFLs, 128GB memory and two Linux guests. Each Linux guest had 4 SAP application server instances, and 8 batch jobs per SAP instance.

The baseline configuration used four 10GbE OSA interfaces, to minimize network contraint, with SMT [5] disabled and enabled. Furthermore, each OSA connection was port-sharing so that the network traffic between the SAP application and database servers would not go outboard of Z to the external LAN. Then the test configuration was switched to a single HiperSockets interface with SMT disabled and enabled for comparison.

The SMT feature is available in IBM Z processors for workloads running on IFLs. SAP application server on Z is an excellent candidate for employing IFL processors and can exploit SMT to gain potential performance benefits. The performance benefits of SMT can vary depending on the characteristics of the workload.

Figure 3 shows the measurement results in batch elapsed times. It shows that HiperSockets (green) is faster than OSA (blue), by 2% to 4%. When SMT was enabled, the batch elapsed time increased by about 30%. The later section 5.3 and 5.4 investigated this performance issue.

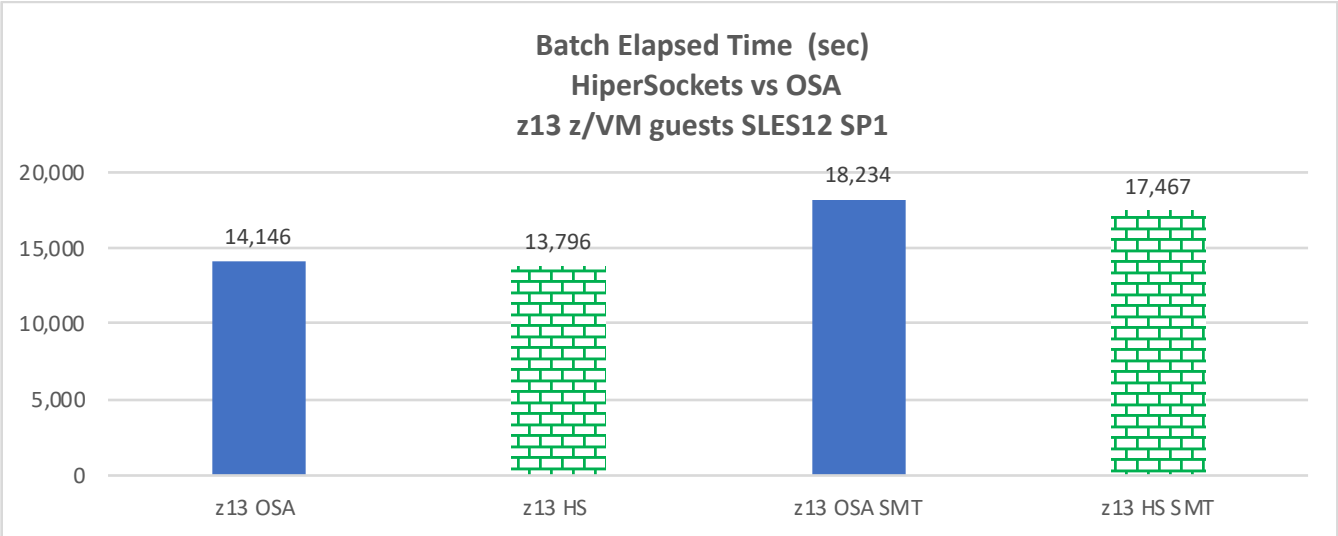

Figure 3: Elapsed Times for HiperSockets vs OSA

Figure 4 shows ITR for the application servers. ITR provides reliable basis for measuring processor capacity. When SMT is enabled, the processing capacity increases. The application server ITR increased by about 50%, a significant gain.

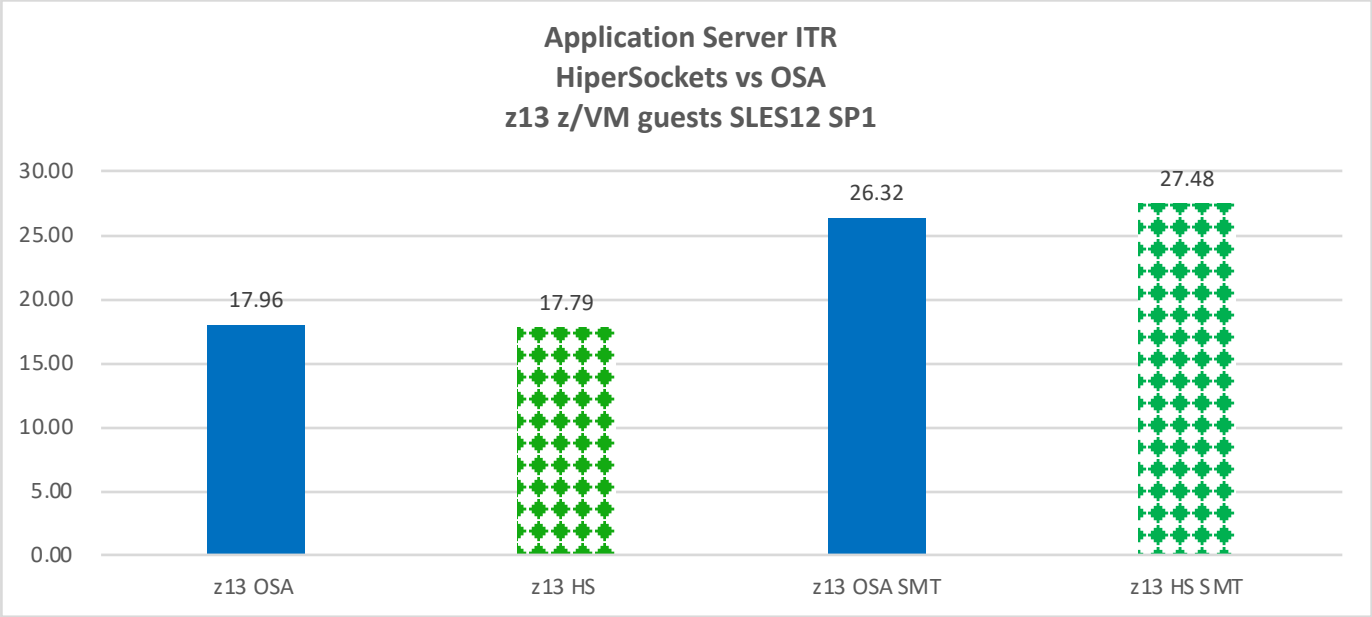

Figure 4: Application Server ITR for HiperSockets vs OSA

As seen previously, when SMT was enabled, the batch elapsed time was elongated. The application server CPU utilization was stuck below 50% (Figure 5). This was an indication of a potential performance bottleneck although there was an increase in processor capacity (Figure 4). As mentioned earlier, the later section 5.3 and 5.4 investigated this performance issue.

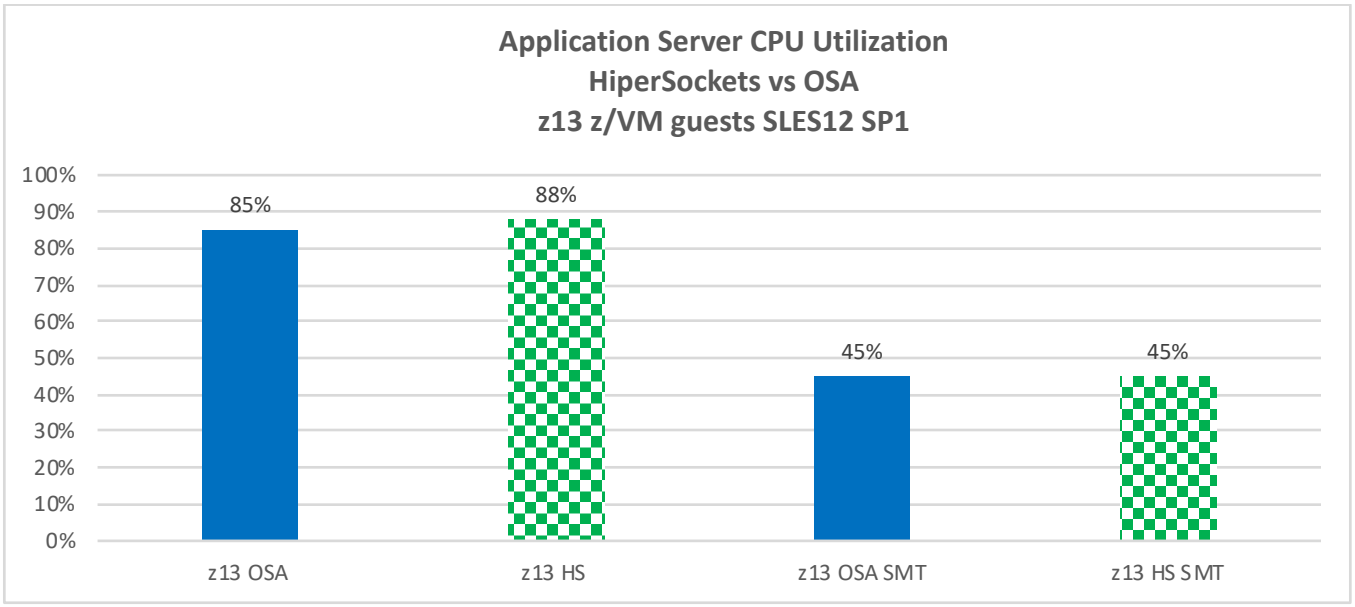

Figure 5: Application Server CPU for HiperSockets vs OSA

5.2 z14 vs z13

The test scenario focus here is the new IBM Z processor, namely z14. It has faster processor cores than z13. Comparison measurements were conducted to assess the z14 performance advantage.

Similar to the previous test configuration, the SAP application server configuration had 4 SLES12 SP1 Linux servers as z/VM guests, running on z14 with 64 IFLs. Two z/VM LPARs were configured. Each z/VM had 32 IFLs, 128GB memory and two Linux guests. Each Linux guest had 4 SAP application server instances, and 8 batch jobs per SAP instance.

For this and subsequent test configurations, only HiperSockets (HS) connection was used, for its enhanced network performance.

Figure 6 shows that z14 (green) has faster batch elapsed time than z13 (blue) by 6% to 8%. When SMT was enabled, batch elapsed time increased by about 30%. The succeeding section 5.3 and 5.4 investigated this performance issue.

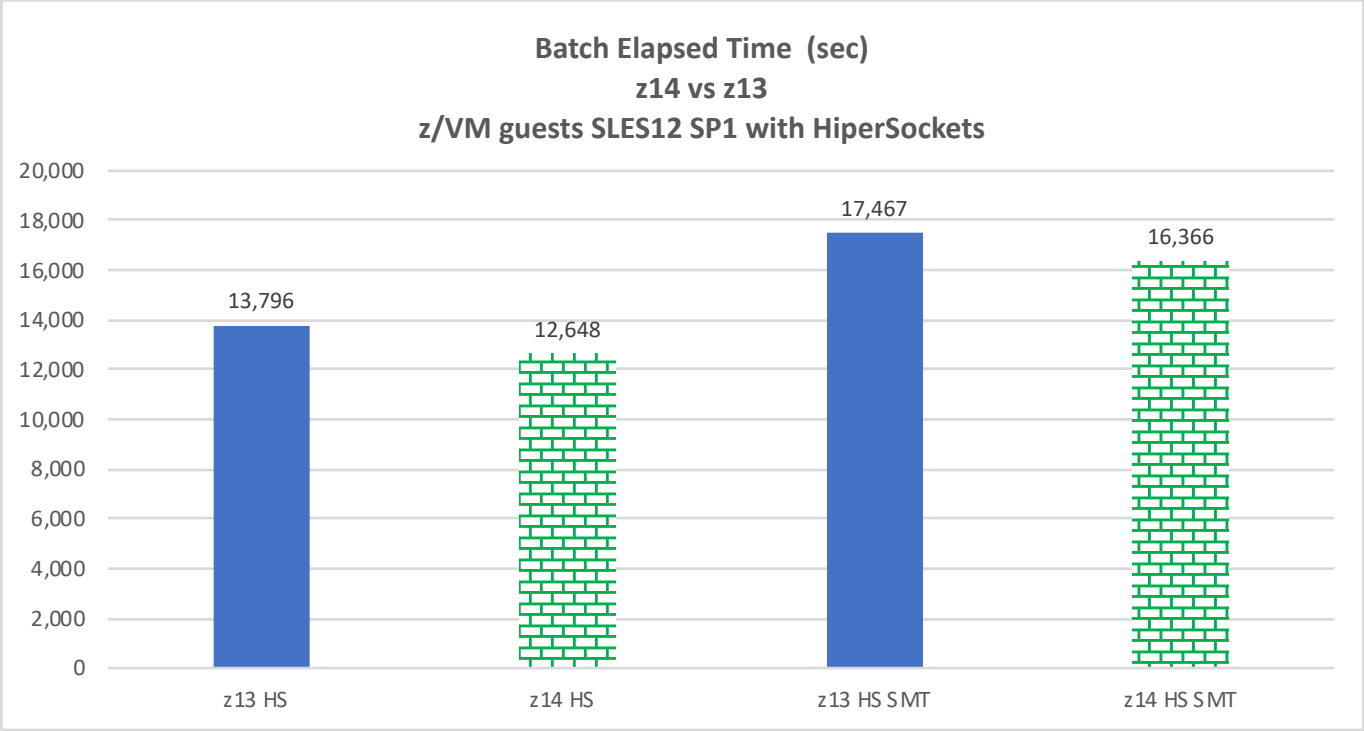

Figure 6: Elapsed Time for z14 vs z13

The following figure shows that the application server ITR is higher on z14 than z13, regardless of the SMT setting. For single-thread core, the ITR improvement of z14 with respect to z13 is 8%. Note that when SMT was enabled, the application server ITR increased further, meaning that the processor capacity also gained. The ITR gain on z13 and z14 with SMT enabled is about 50% for this workload. In general, SMT technology provides a similar boost for z13 and z14.

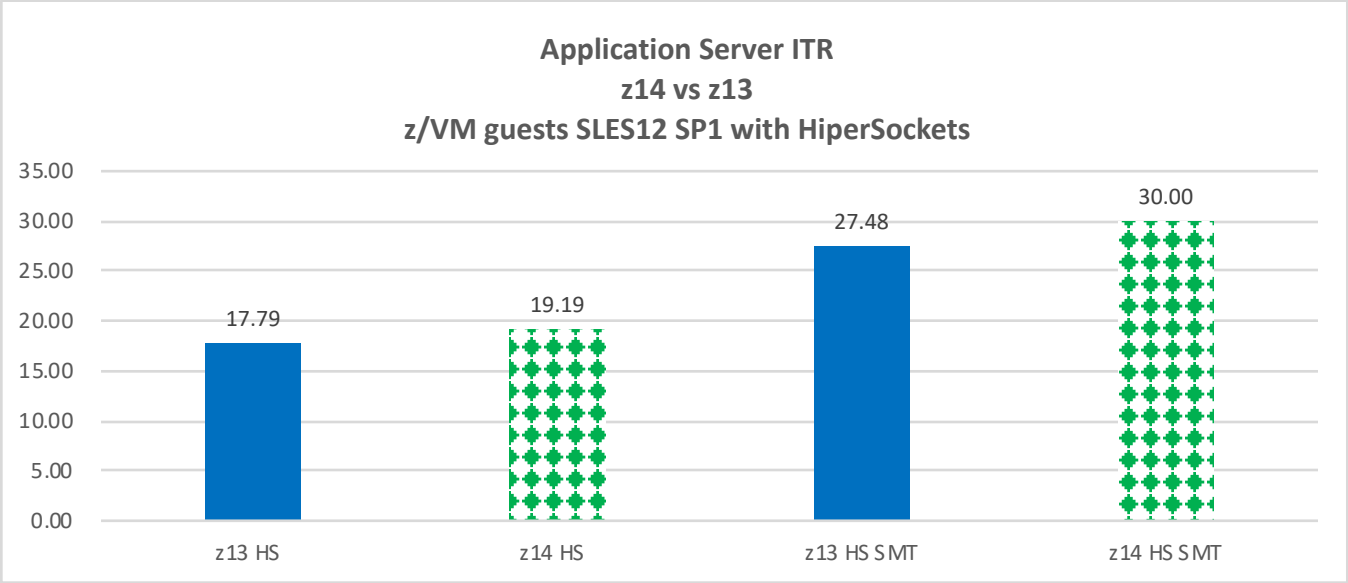

Figure 7: Application Server ITR for z14 vs z13

Figure 8 shows that the database server ITR is higher on z14 (green) than z13 (blue) by 9%. Note that the database server was in a z/OS LPAR with 16 dedicated general processors which were not eligible for SMT exploitation. The result here is rather a performance capacity view of the SAP database servers on two recent Z machines.

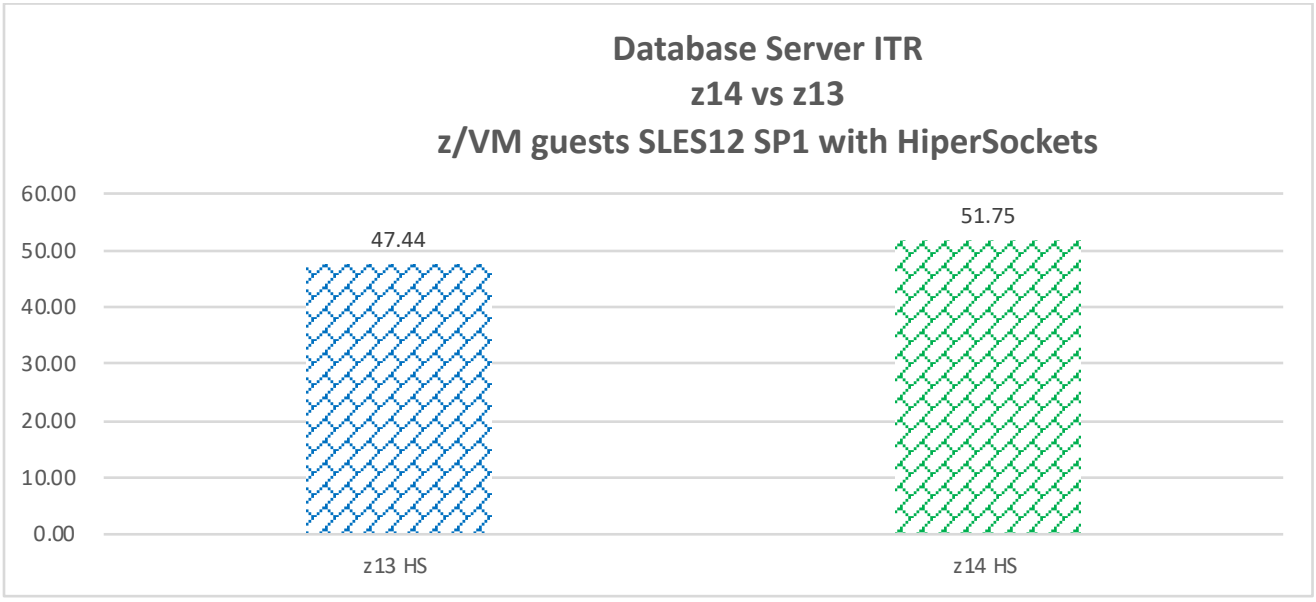

Figure 8: Database Server ITR for z14 vs z13

Figure 9 shows that when SMT was enabled on the application server, the elapsed time was elongated. The application server CPU utilization was stuck below 50%. This was an indication of a potential performance bottleneck although there was an increase in processor capacity as shown in Figure 7. The next two sections investigated this performance issue.

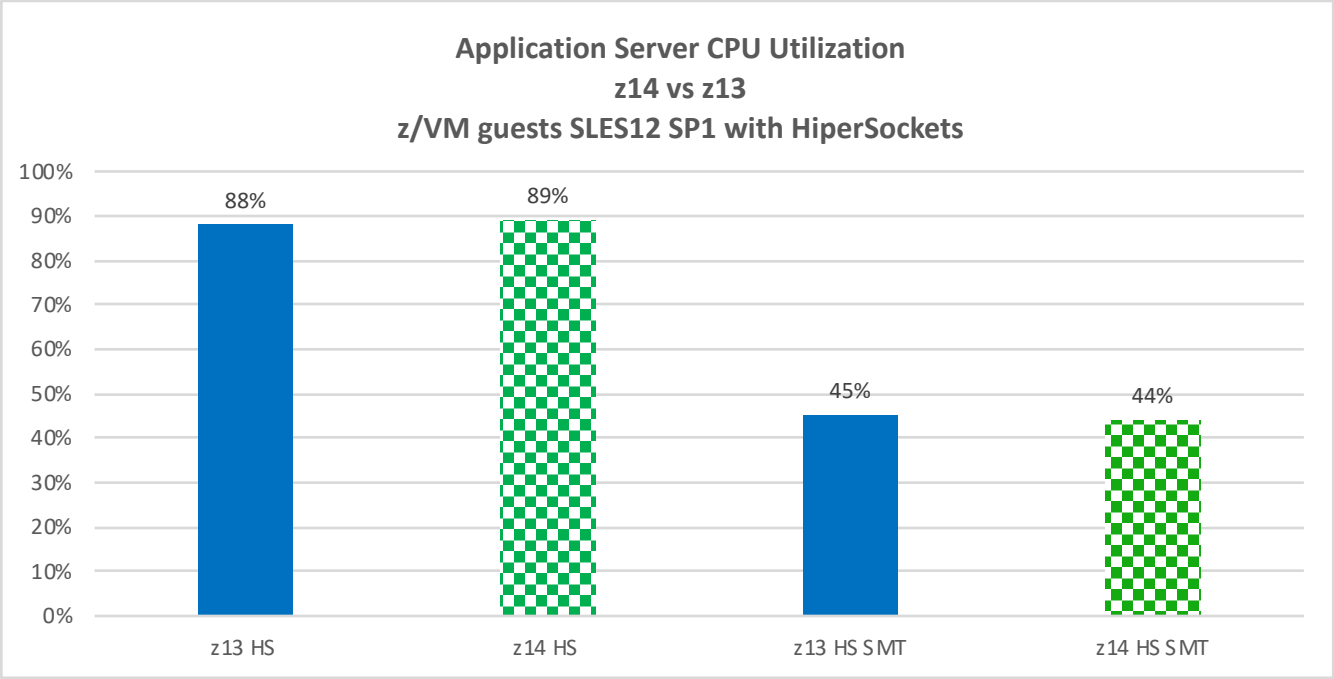

Figure 9: Application Server CPU Utilization for z14 vs z13

Thus far the study shows that z14 is faster than z13 in batch elapsed time and provides more processor capacity for the application and database servers, regardless of the SMT setting. Hereon the focus will be on z14, thus only z14 is used in the subsequent test configurations.

5.3 Single z/VM

In the previous measurement results, the application server CPU utilzation was stuck under 50% with SMT enabled, indicating a potential bottleneck. This measurement scenario was to investigate if the SMT bottleneck would be related to the physical 64 processor thread limit per z/VM.

The test configuration for the application servers had been using four Linux guests on two z/VM systems. This measurement scenario was a single z/VM image with 64 IFLs, to be compared to two z/VM images with 32 IFLs each. In this case, SMT was disabled since z/VM supports up to a maximum of 64 physical processor threads per LPAR image. Increasing the number of z/VM guests (8, 16, or more) had been considered to evaluate the z/VM guest scaling effect. Due to the time and resource constraint, the z/VM guest scaling evaluation will be considered in a future study.

Figure 10 shows that the batch elapsed times are comparable for both z/VM configuration scenarios. There is no constraint running on a single z/VM LPAR with 64 physical processor threads.

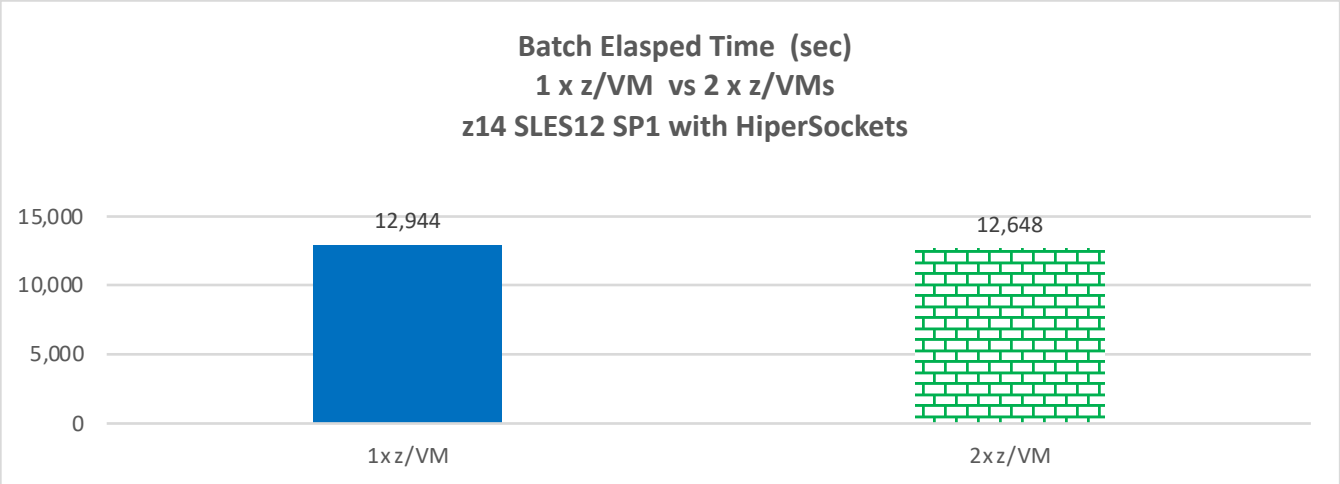

Figure 10: Elapsed Times for 1 z//VM vs 2 z/VMs

Figure 11 shows that the application server processor utilization is not limited, indicating that there is no obvious performance constraint, in either z/VM configuration scenario.

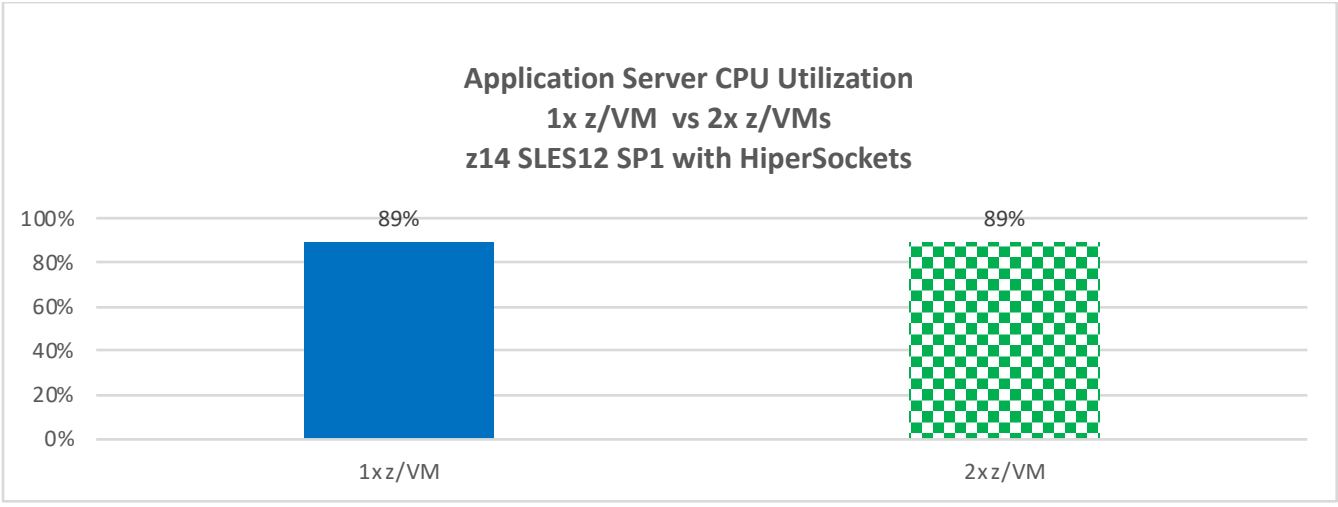

Figure 11: Application Server CPU for 1 z//VM vs 2 z/VMs

5.4 Topology Patch for z/VM Linux Guests

The test configuration for the application servers has been running under z/VM as guests. One complication of Linux running under z/VM is that the guest is not aware of the underlying CPU topology, which may result in dispatching inefficiency by the Linux scheduler. This is not an issue for native Linux. This is a general issue for Linux guest running under a virtualization layer regardless on any platform and technology

To address this issue, a Linux kernel “topology patch” was implemented to allow switching the CPU topology under z/VM to "all CPUs are siblings" (sw_siblings). WIth the sw_siblings setting the scheduler will move a freshly awakened task to any virtual CPU that is idle. This topology patch is relevant only to the z/VM Linux guest configuration.

Figure 12 shows that the batch elapsed time has significant improvement with the topology patch when SMT is enabled. Essentially, applying the topology patch (green) eliminates the previously mentioned of about 30% increase of batch elapsed time (blue) when SMT is enabled.

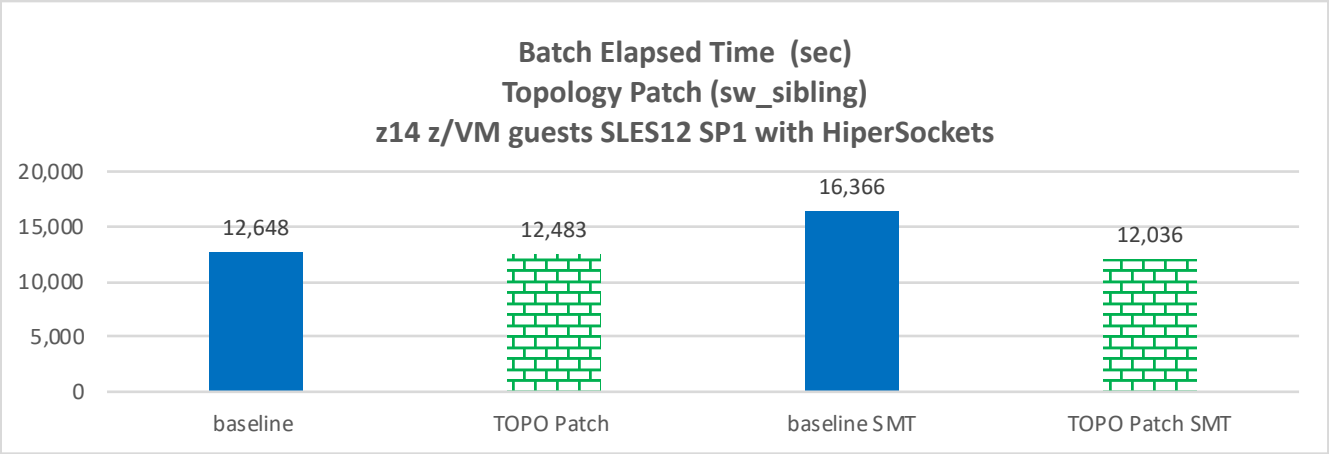

Figure 12: Elapsed Times With vs. Without Topology Patch

Figure 13 shows that with the topology patch, the application server CPU utilization is not limited with SMT, resulting in significant improvement of the batch elapsed time. The previously mentioned performance constraint was alleviated with the topology patch enabled.

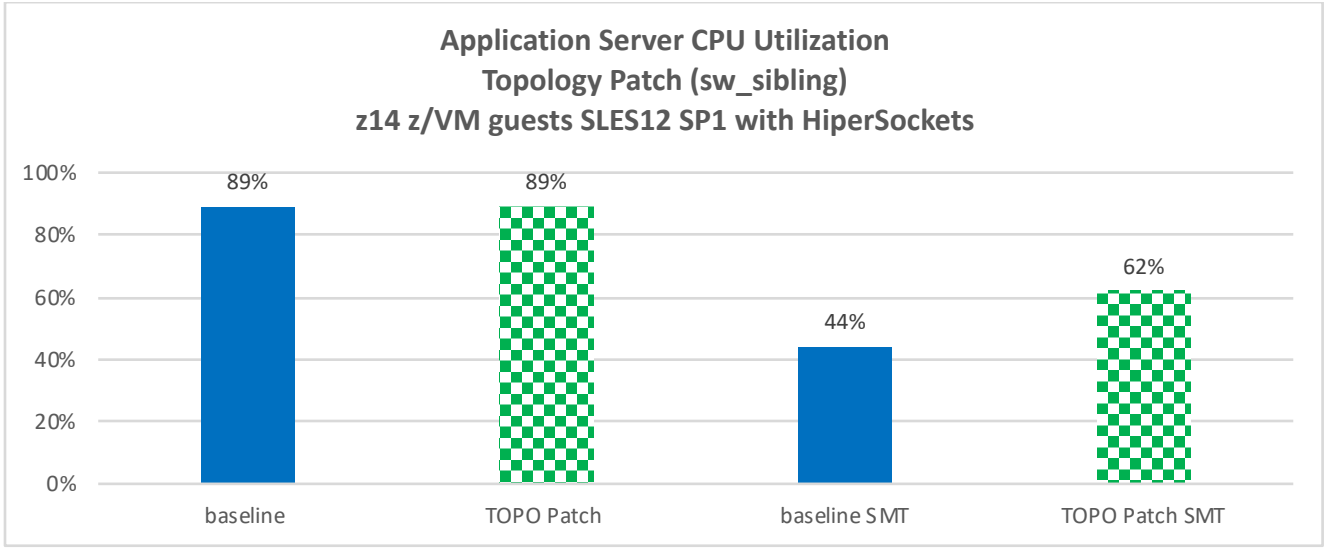

Figure 13: Application Server CPU Utilization With vs. Without Topology Patch

5.5 Native Linux vs z/VM Linux Guest

The test scenario focus here is running the application server on native Linux. Unlike the z/VM Linux guest, the native Linux is aware of the underlying CPU topology. It does not need the topology patch, as discussed in the preceding section. The study explored the performance difference of running the SAP application server in native Linux versus as Linux guest under z/VM with the topology patch scenario. Running the SAP application server as z/VM Linux guest with topology patch had the best batch elapsed time among all the measurements so far.

Figure 14 shows that the batch elapsed time for Native Linux (green) is less than the time for z/VM Linux guest (blue) by up to 20% with SMT enabled. It also indicates that running native Linux with SMT enabled is the fastest configuration.

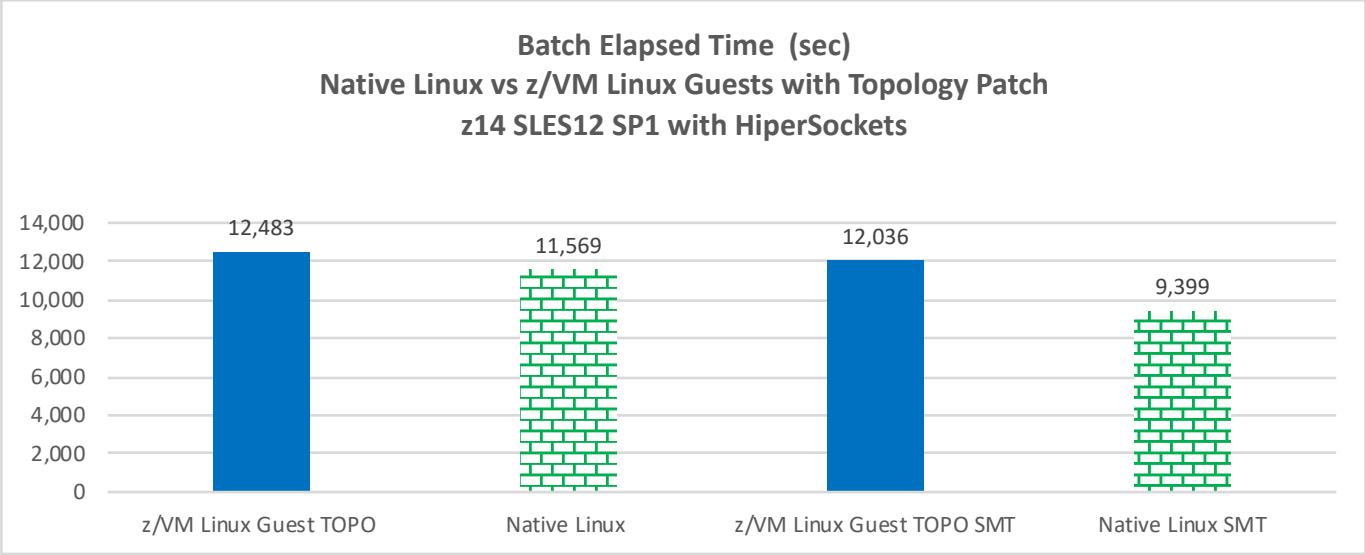

Figure 14: Elapsed Times for Native Linux vs z/VM Linux Guest

The following figure shows that the application server ITRs are comparable for both z/VM Linux guests and native Linux. This also indicates that there is low CPU overhead for using z/VM Linux guest configuration. Subsequent measurement configuration used native Linux LPAR with SMT for it provided the better batch elapsed time.

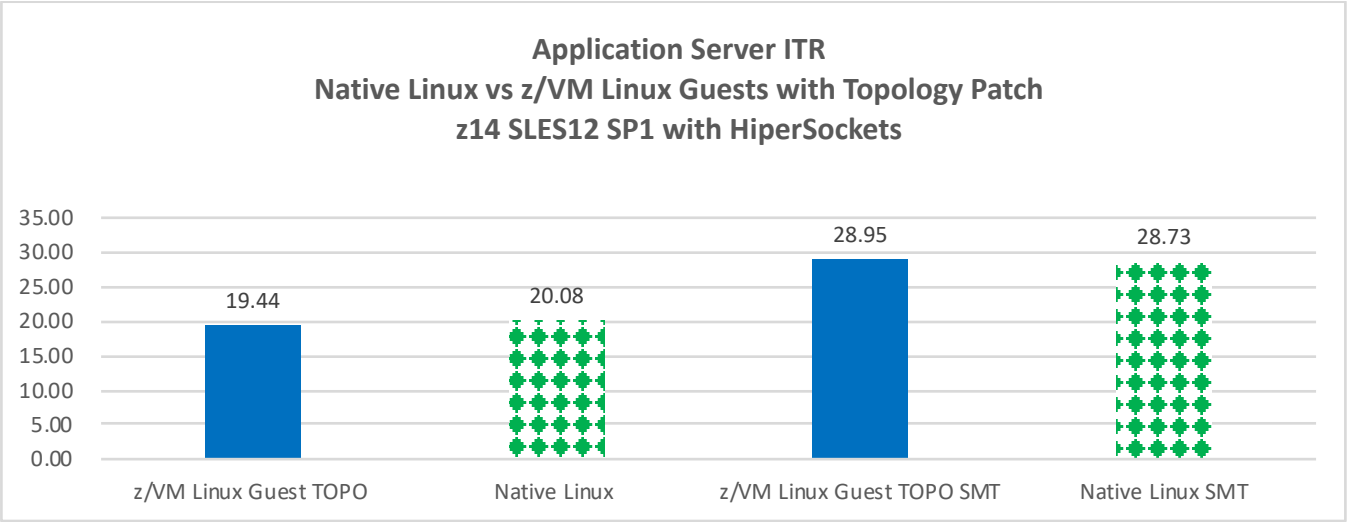

Figure 15: Application Server ITR for Native Linux LPAR vs z/VM Linux Guest

5.6 Vertical CPU Polarization

By default, Linux system on Z sets dispatching mode to horizontal CPU polarization which means all CPUs are dispatched for the same amount of time. If dispatching mode is set to vertical CPU polarization, the hypervisor dispatches certain CPUs longer time than other CPUs. The idea is to get processing done on the same group of CPUs, to preserve cache contents, and to minimizes cache misses and disruption. This translates to increasing processing efficiency and provides potential performance benefit.

The following commands are examples for changing polarization to vertical:
 # chcpu -p vertical

Alternatively,
 # echo "1" > /sys/devices/system/cpu/dispatching

The following is an example of polarization state before and after the switch. Note that there are three types of vertical CPUs: high, medium, and low. Low CPUs hardly get any real CPU time, while high CPUs get a full real CPU. Medium CPUs get something in between.

```
lx014:~ # 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    1      1    2:2:2:2         yes    yes        horizontal   2
3   0    0      0    1      1    3:3:3:3         yes    yes        horizontal   3
4   0    0      0    1      2    4:4:4:4         yes    yes        horizontal   4
5   0    0      0    1      2    5:5:5:5         yes    yes        horizontal   5
6   0    0      0    1      3    6:6:6:6         yes    yes        horizontal   6
7   0    0      0    1      3    7:7:7:7         yes    yes        horizontal   7
lx014:~ # chcpu -p vertical
Successfully set vertical dispatching mode
lx014:~ # 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    1      1    2:2:2:2         yes    yes        vert-medium  2
3   0    0      0    1      1    3:3:3:3         yes    yes        vert-medium  3
4   0    0      0    1      2    4:4:4:4         yes    yes        vert-medium  4
5   0    0      0    1      2    5:5:5:5         yes    yes        vert-medium  5
6   0    0      0    1      3    6:6:6:6         yes    yes        vert-low     6
7   0    0      0    1      3    7:7:7:7         yes    yes        vert-low     7
```

When an image is defined as shared, there will be a combination of vertical high, medium and low CPUs assigned to the image dependent on the weight for the image with respect to the resources available and the other image definitions in the CEC. If the image is designated as dedicated, there will only be vertical CPUs allotted to the image. This test environment had only one type, all vertical high CPUs, by configuring processors as dedicated, not shared. Also, note that the vertical CPU polarization setting is relevant only to native Linux, and not applicable to z/VM Linux guests.

The test configuration had 4 native Linux LPARs with SMT enabled, 16 IFLs and 64 GB memory each. The following figure shows that the batch elapsed times are equivalent for the baseline and vertical CPU polarization scenarios.

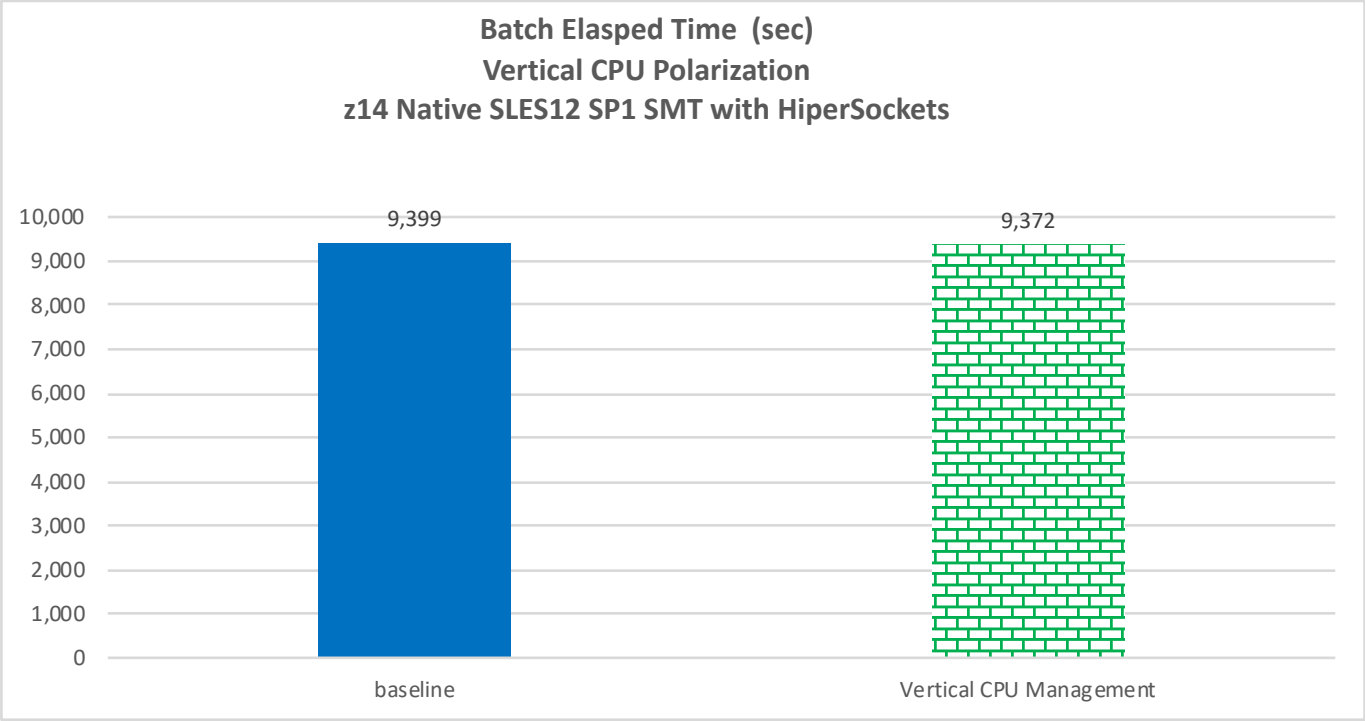

Figure 16: Elapsed Times for Vertical CPU Polarization vs Baseline

5.7 Real-Time Scheduling Policy

The Linux scheduler handles CPU resource allocation for executing threads or processes with the aim of maximizing performance. It is responsible for determining when and how long a thread runs on a particular CPU core, with real-time and normal scheduling policies. Real-time threads are scheduled first because they are used for time-critical tasks. Normal threads are then scheduled after the real-time threads are scheduled.

This study explored potential performance improvement with Linux real-time scheduling policy, specifically round-robin scheduling (SCHED_RR). With SCHED_RR, each process (ie, SAP PID) can be scheduled with a priority value that ranges from 1 to 99. For the comparison, the batch task priority was set to 99, for the highest priority.

There are SAP profile parameters for these settings:
rdisp/linux/os_scheduler = SCHED_RR
rdisp/linux/os_priority =99.

Figure 17 shows that the batch elapsed time for real-time scheduling policy (green) is 6% faster than the baseline (blue). It is a promising tuning parameter.

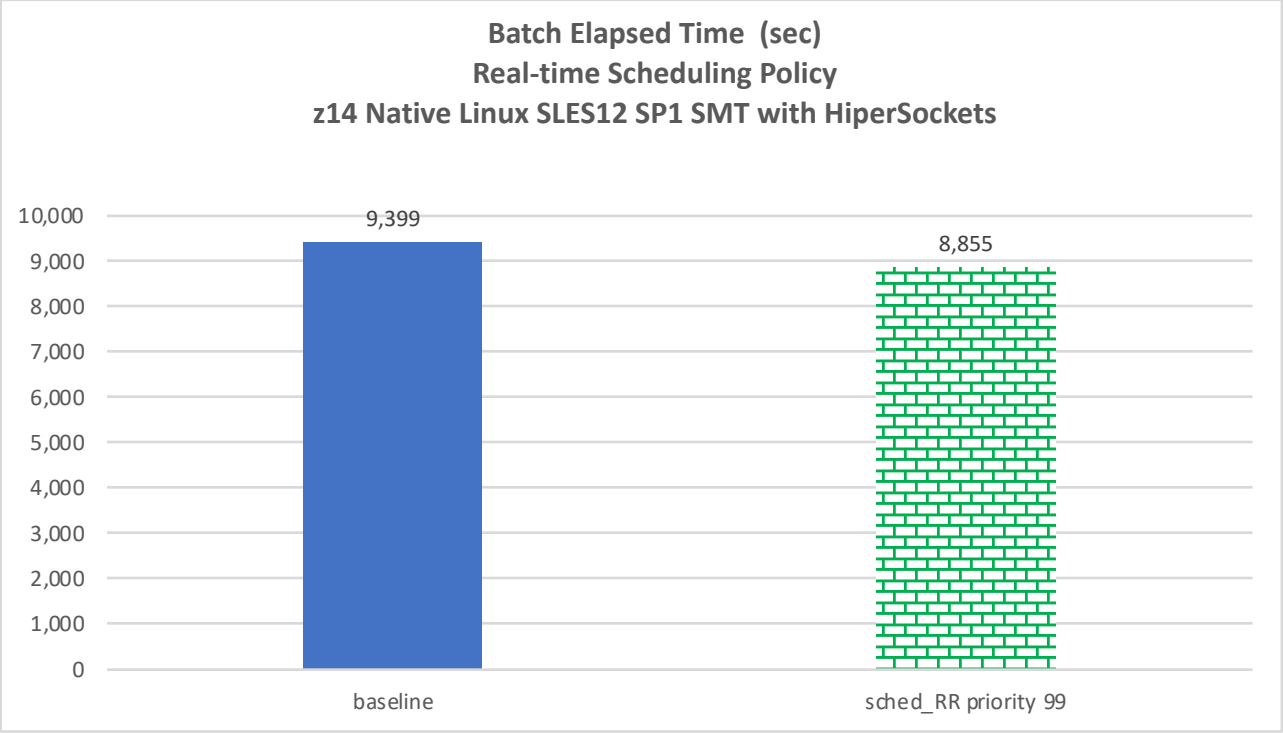

Figure 17: Elapsed Times for Real-time Scheduling Policy vs Baseline

5.8 Receive Packet Steering (RPS)

Network technology has advanced greatly and can move enough network traffic data to the point that the limiting factor to achieving maximum performance is at the host system as the CPU clock frequency has not continued to improve at an equivalent rate. In order to keep up, computer systems have shifted to designs with more CPU cores.

One way to distribute data packets across multiple CPUs so that they can be processed in parallel is to use Receive Packet Steering (RPS). It uses a hash of the IP addresses and port numbers. This hash is to ensure that data packets for the same stream of data are sent to the same CPU. This helps increasing performance.

This study investigated settings for all CPU cores, every 2nd core, and every 4th core. The test network interface used is a single HiperSockets interface. It has 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 bitmap needs to be set to 1.

The following example illustrates before and after enabling RPS setting for all 16 CPU cores for a HiperSockets interface hsi0.

```
Default RPS setting:
# cat /sys/class/net/hsi0/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000


To enable RPS setting for all 16 CPU cores:
#  echo "0000ffff" > /sys/class/net/hsi0/queues/rx-0/rps_cpus


RPS setting enabled for 16 CPU cores:
# cat /sys/class/net/hsi0/queues/rx-0/rps_cpus
00000000,00000000,00000000,00000000,00000000,00000000,00000000,0000ffff
```

Figure 18 shows that the batch elapsed times for these various RPS settings against the baseline. Although the best result is RPS with every 4th CP cores, implementing with all CP setting is simpler than implementing for either every 2nd or 4th CPs or any other variance. The all CP setting yielded the bulk of the elapsed time improvement from the baseline; whereas the other settings provided small incremental improvement. As a result, for ease of implementation, all CP setting is preferable.

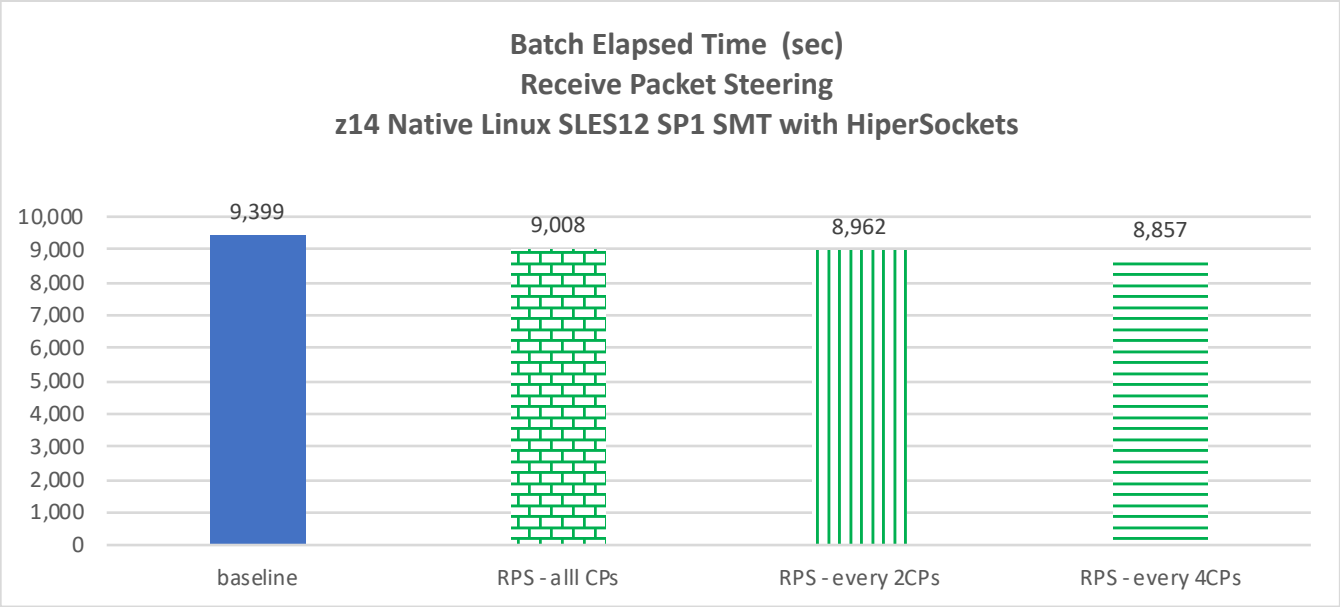

Figure 18: Elapsed Times for RPS Scenarios vs Baseline

Figure 19 shows that the application server ITRs were comparable. The various RPS settings do not have significant CPU impact.

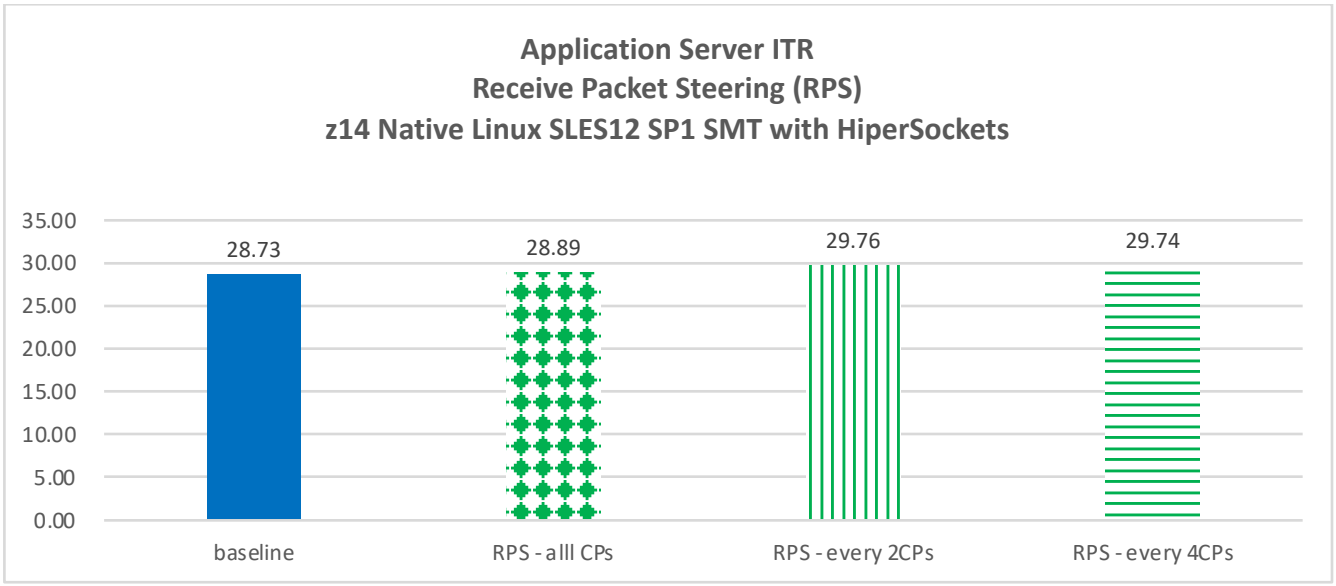

Figure 19: Application Server ITR for RPS Scenarios vs Baseline

5.9 SLES12 SP3 vs SP1

SLES12 SP1 was used for the SAP application server operating system. This study investigated if there is any performance difference by upgrading to the more recent SLES12 SP3.

Figure 20 shows that the batch elapsed times are equivalent for SLES12 SP3 and SP1.

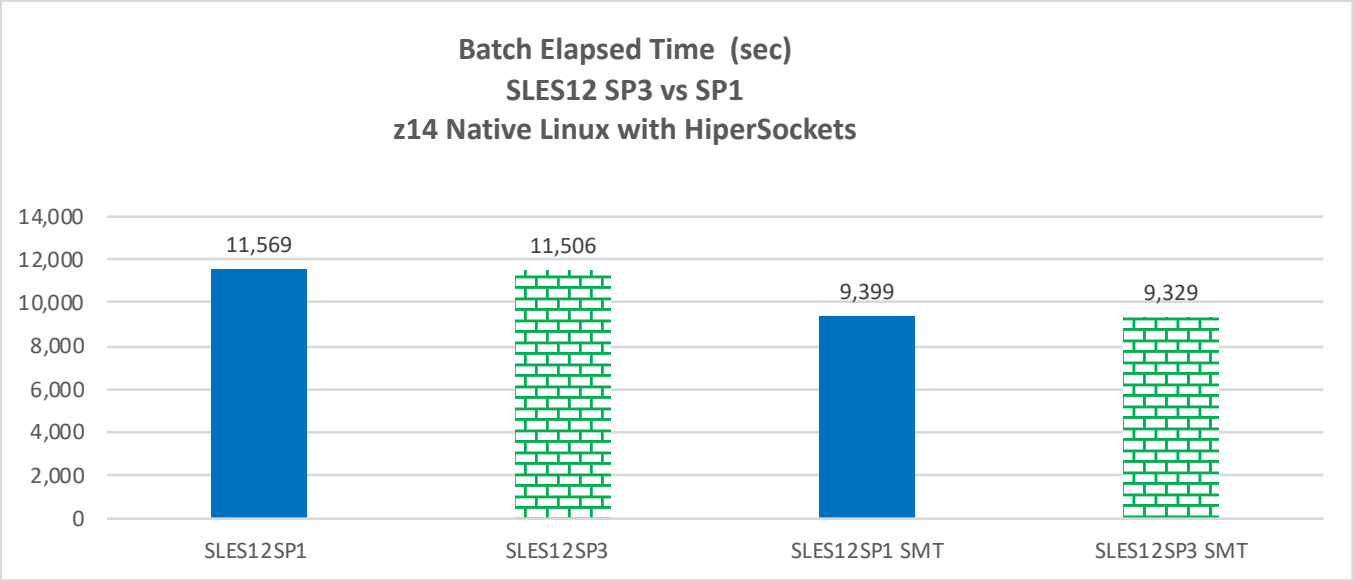

Figure 20: Elapsed Times for SLES12 SP3 vs SP1

Figure 21 shows that the application server ITRs are also equivalent for the SLES12 SP3 and SP1. Upgrading to the SP3 release does not pose a performance issue for the SAP banking account settlement.

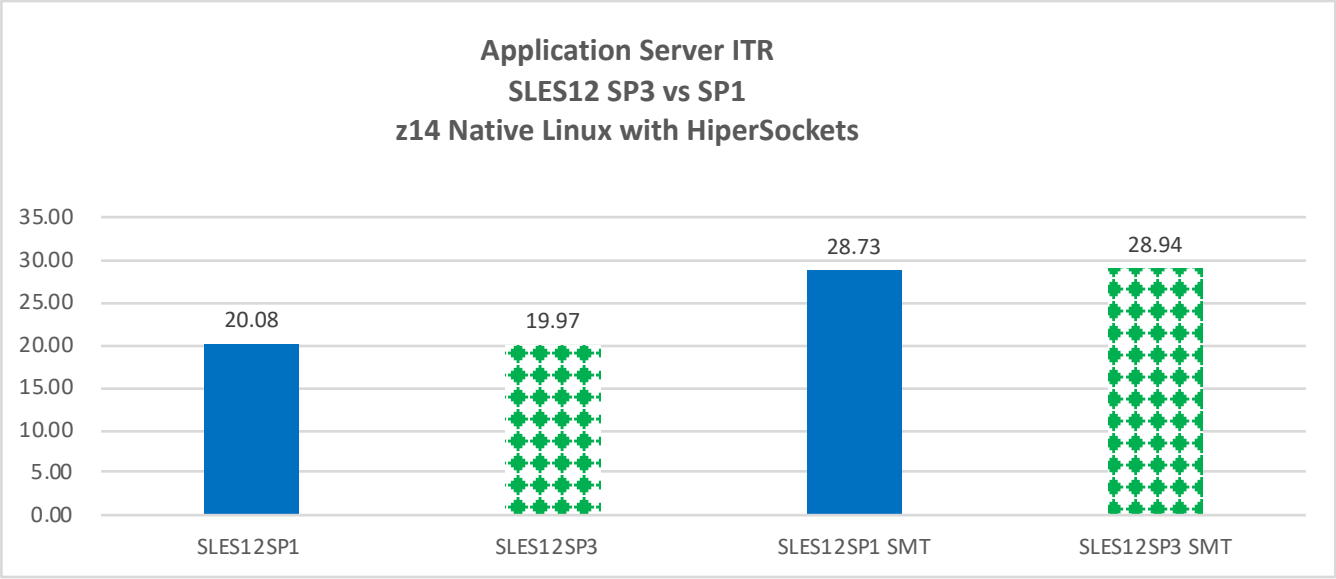

Figure 21: Application Server ITR for SLES12 SP3 vs SP1

5.10 Increase Size of Application Server

The size of the application server processor configuration had been kept constant at 64 IFLs. To further shorten the batch elapsed time, one simple way is to increase the processor cores for the application servers. This study investigated performance scaling by doubling the IFL cores from 64 to 128 to match the 128 parallel batch jobs. In this case, the 128 jobs would be dispatched to the 128 IFLs concurrently without SMT enabled.

The IBM z14 3906-M05 used in this test environment can have more than 128 IFLs. More IFLs can be added and SMT can be enabled to further increase the processor capacity for the application server. However, the number of batch jobs would need to be increased and the test environment would need to be tuned further to take full advantage of the additional processor threads and capacity. Due to time and resource constraints, this is beyond the scope of this study.

Figure 22 shows that the batch elapsed time is significantly improved with 128 IFL processor configuration without SMT. The time improved 34% as compared to the 64 IFL processor configuration with SMT disabled and 18% as compared to the 64 IFL configuration with SMT enabled.

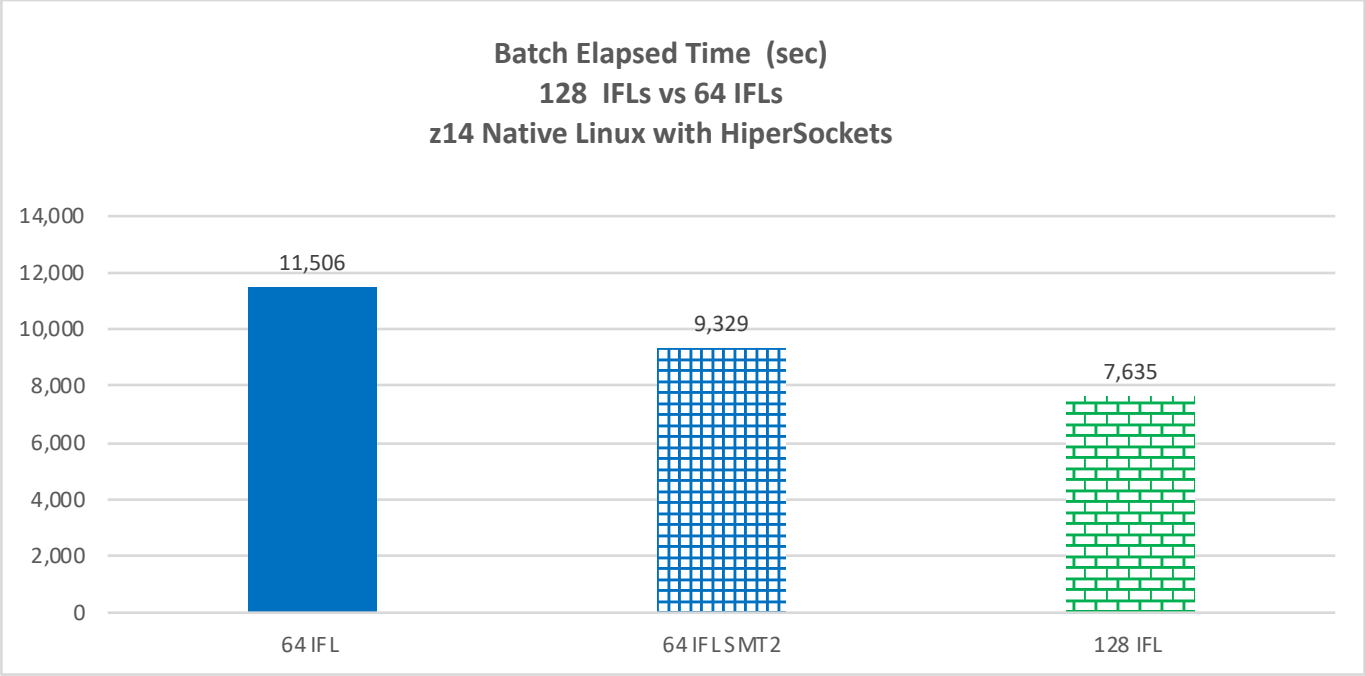

Figure 22: Elapsed Times for the Application Server at 128 vs 64 IFLs

5.11 Best tuning parameters on 128 IFLs

Earlier experiments with Linux tuning parameters SCHED_RR and RPS showed improvement in the batch elapsed times. Here they were revisited and applied to the measurement configuration at 128 IFL processors with SMT disabled.

The following figure shows that the batch elapsed time is best for RPS among the three scenarios. SCHED_RR did not provide an improvement for the 128 IFL configuration although it was better than the baseline at 64 IFLs, as discussed in "*Real-Time Scheduling Policy (section 5.7)"*. RPS is consistently better for both 64 and 128 IFLs.

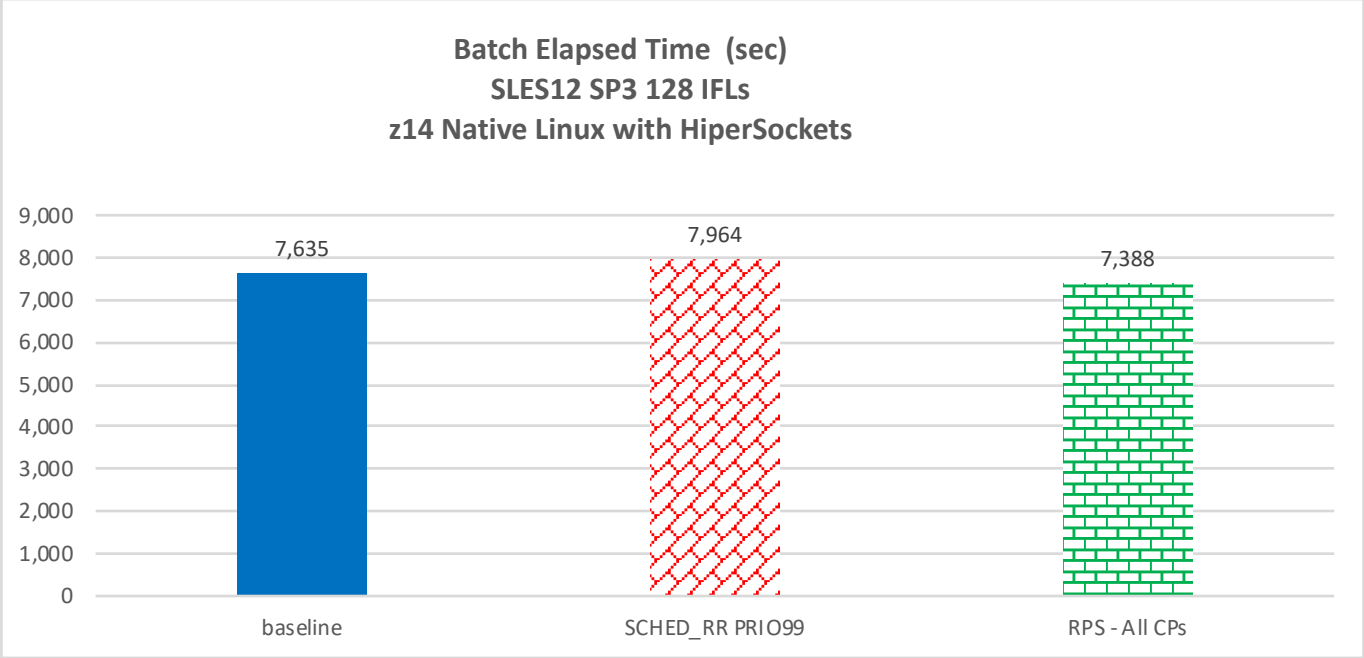

Figure 23: Elapsed Times for Best Tuning Parameters on 128 IFLs

6.0 Conclusion

This study has taken a holistic approach and it is the first of its kind within IBM Z performance. A series of large-scale measurements with a customer-like multi-tier workload was conducted to determine performance benefits and advantages of various colocated configurations. The following figure shows a subset of the measurements processing the SAP Banking Account Settlement with 60 million accounts. It illustrates that with proper configuration and features the batch elapsed time can be shortened dramatically. In this case the time went from above 5 hours to just about 2 hours!

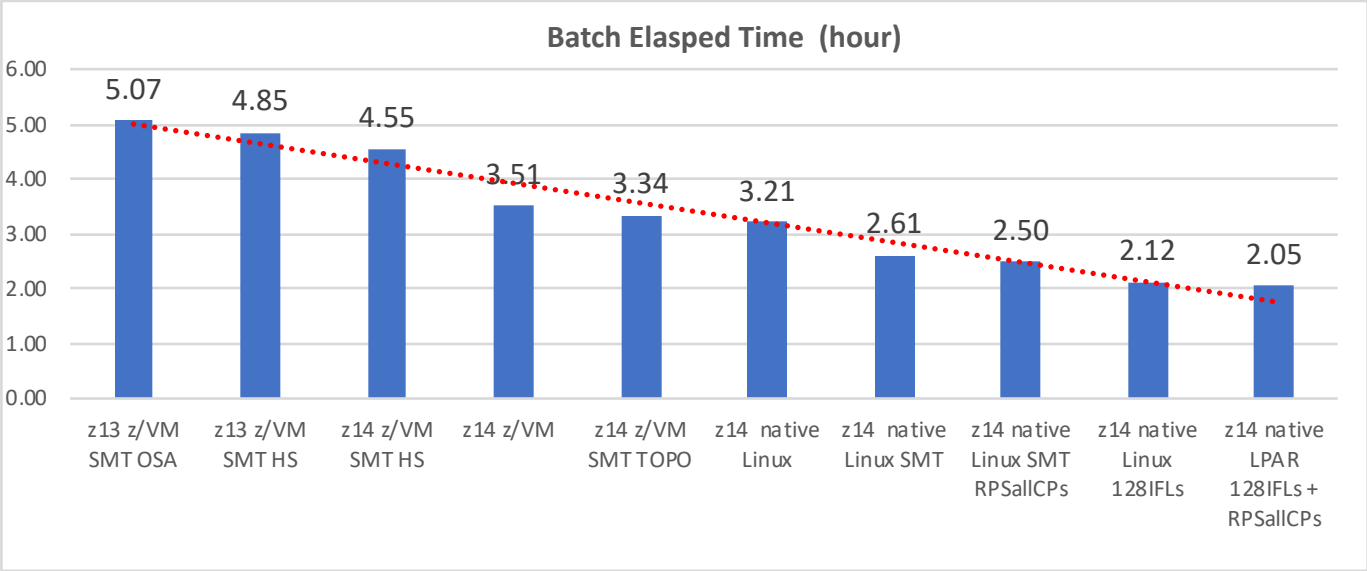

Figure 24: Trendline of Batch Elapsed Times

The ability to process 60 million bank accounts in about 2 hours demonstrates that IBM Z can process a huge number of accounts for a very large bank within a critical batch window. IBM Z can help banking customers to meet their business and regulatory requirements.

The measurement results of this study with SAP banking Account Settlement workload has shown the following performance improvements which may vary for different workload and environment:

- **HiperSockets** provides faster network communication than OSA and speeds up the batch elapsed time by 2-4%.
- **z14** has faster processor cores than z13 and reduces the batch elapsed time by 6-8%.
- **Native Linux** without the z/VM layer shortens the batch elapsed time by 7-22%.
- Among the three Linux tuning parameters:
  - Vertical CPU Polarization has marginal (<1%) improvement.
  - Real-Time Scheduling (SCHED_RR) has mixed results.
  - **Receive Packet Steering** (RPS) cuts the batch elapsed time by 4-6%. The best result was RPS with every 4th CP core among the 3 different settings experimented. Other variance may have further improvement. However, all CP setting is preferable for ease of implementation.
- Adding more processor capacity by **increasing number of cores** and/or enabling **SMT** can further improve the batch elapsed time as shown by the cases of 128 IFLs compared to 64 IFLs.

IBM z14 provides up to 170 cores. The batch elapsed time could be shortened further by adding IFLs and enabling SMT so to increase the processor capacity for the Linux application server. If given more time and resources, the number of batch jobs can be adjusted with further test environment tunings to take full advantage of the additional processor threads and capacity.

In summary, recent Z machine generations with more processor cores can host very large SAP workloads in a single Z CEC and provide a viable platform for customers to implement server consolidation and colocation. IBM Z has capabilities to process data extremely fast by using I/O features to improve the latency of data access and large memory to process large volumes of data. IBM Z is built for robustness, security, scalability, and business continuity. IBM Z can meet all the stringent criteria to handle the demanding large-scale SAP Banking production workloads [6].

7.0 Appendix

The summary data of the measurements discussed in this paper are listed in the following tables:

runid	S80206B1	S80214B1	S80130B1	S80122B1
Processor (DB server)	z13-16w	z13-16w	z13-16w	z13-16w
APP server O/S	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1
App server config	2x z/VM	2x z/VM	2x z/VM	2x z/VM
Processor (App servers)	64 IFL	64 IFL	64 IFL	64 IFL
SMT	no	yes	no	yes
Additional variables	4 OSA	4 OSA	HiperSockets	HiperSockets
# of jobs	128 jobs	128 jobs	128 jobs	128 jobs
z/OS Utilization	31%	24%	33%	26%
App server Utilization	85%	45%	88%	45%
Total # of Accounts (60M)	60,000,000	60,000,000	60,000,000	60,000,000
Batch elasped time (sec)	14,146	18,234	13,796	17,467
Avg SAP App CPU time (sec)	5,862.64	7,943.18	5,891.21	7,914.70
Avg time spent in Db2 (sec)	581.14	580.01	576.53	580.29
Avg Db2 CPU Time (sec)	396.32	390.56	394.16	388.39
Avg DB2 client/server latency (usec)	1,482	1,691	1,379	1,516
ETR (Million accounts per hour)	15.27	11.85	15.66	12.37
ITR (DB server)	49.26	49.36	47.44	47.56
ITR (App server)	17.96	26.32	17.79	27.48
Avg Db2 Logging Rate (MB/sec)	178.30	138.30	182.88	144.36

Table 1: z13 HiperSokets & OSA

runid	S70117B1	S71203B1	S80128B2	S80123B1	S71222B1
Processor (DB server)	z14-16w	z14-16w	z14-16w	z14-16w	z14-16w
APP server O/S	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1
App server config	2x z/VM	2x z/VM	1x z/VM	2x z/VM	2x z/VM
Processor (App servers)	64 IFL	64 IFL	64 IFL	64 IFL	64 IFL
SMT	no	yes	no	no	yes
Additional variables				Topology patch	Topology patch
# of jobs	128 jobs	128 jobs	128 jobs	128 jobs	128 jobs
z/OS Utilization	33%	25%	32%	33%	34%
App server Utilization	89%	44%	89%	89%	62%
Total # of Accounts (60M)	60,000,000	60,000,000	60,000,000	60,000,000	60,000,000
Batch elasped time (sec)	12,648	16,366	12,944	12,483	12,036
Avg SAP App CPU time (sec)	5,500.61	7,176.86	5,633.05	5,458.47	7,250.76
Avg time spent in Db2 (sec)	523.14	519.17	521.98	521.40	522.86
Avg Db2 CPU Time (sec)	345.99	342.60	344.42	345.94	346.37
Avg DB2 client/server latency (usec)	1,184	1,401	1,183	1,096	686
ETR (Million accounts per hour)	17.08	13.20	16.69	17.30	17.95
ITR (DB server)	51.75	52.79	52.15	52.43	52.78
ITR (App server)	19.19	30.00	18.75	19.44	28.95
Avg Db2 Logging Rate (MB/sec)	199.50	154.42	194.82	202.20	210.14

Table 2: z14 z/VM Linux Guests & Topology Patch

runid	S71208B1	S71219B1	S80112B1	S80114B1	S80115B1
Processor (DB server)	z14-16w	z14-16w	z14-16w	z14-16w	z14-16w
APP server O/S	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1	SLES 12 SP1
App server config	4x LPAR	4x LPAR	4x LPAR	4x LPAR	4x LPAR
Processor (App servers)	64 IFL	64 IFL	64 IFL	64 IFL	64 IFL
SMT	yes	yes	yes	yes	yes
Additional variables	Vertical CPU Polarization	SCHED_RR	RPS - All CPs	RPS - every 2CPs	RPS - every 4CPs
# of jobs	128 jobs	128 jobs	128 jobs	128 jobs	128 jobs
z/OS Utilization	44%	47%	46%	46%	47%
App server Utilization	79%	82%	83%	81%	82%
Total # of Accounts (60M)	60,000,000	60,000,000	60,000,000	60,000,000	60,000,000
Batch elasped time (sec)	9,372	8,855	9,008	8,962	8,857
Avg SAP App CPU time (sec)	7,355.62	7,212.87	7,288.87	7,232.92	7,253.19
Avg time spent in Db2 (sec)	544.03	565.76	551.40	547.74	550.14
Avg Db2 CPU Time (sec)	357.32	357.49	355.69	355.73	355.50
Avg DB2 client/server latency (usec)	298	165	180	265	215
ETR (Million accounts per hour)	23.05	24.39	23.98	24.10	24.39
ITR (DB server)	52.38	51.90	52.13	52.40	51.89
ITR (App server)	29.17	29.75	28.89	29.76	29.74
Avg Db2 Logging Rate (MB/sec)	269.56	284.96	280.24	281.06	284.86

Table 3: z14 Tuning Parameters @ 64 IFLs

runid	S71103B1	S71110B1	S80213B1	S80209B1
Processor (DB server)	z14-16w	z14-16w	z14-16w	z14-16w
APP server O/S	SLES 12 SP1	SLES 12 SP1	SLES 12 SP3	SLES 12 SP3
App server config	4x LPAR	4x LPAR	4x LPAR	4x LPAR
Processor (App servers)	64 IFL	64 IFL	64 IFL	64 IFL
SMT	no	yes	no	yes
Additional variables				
# of jobs	128 jobs	128 jobs	128 jobs	128 jobs
z/OS Utilization	36%	44%	36%	44%
App server Utilization	93%	80%	94%	80%
Total # of Accounts (60M)	60,000,000	60,000,000	60,000,000	60,000,000
Batch elasped time (sec)	11,569	9,399	11,506	9,329
Avg SAP App CPU time (sec)	5,335.40	7,395.68	5,336.93	7,367.67
Avg time spent in Db2 (sec)	533.74	544.61	538.95	539.18
Avg Db2 CPU Time (sec)	354.79	355.44	352.46	353.53
Avg DB2 client/server latency (usec)	1,109	308	1,019	272
ETR (Million accounts per hour)	18.67	22.98	18.77	23.15
ITR (DB server)	51.86	52.23	52.15	52.62
ITR (App server)	20.08	28.73	19.97	28.94
Avg Db2 Logging Rate (MB/sec)	218.14	268.46	219.22	271.12

Table 4: z14 Native Linux, SLES12 SP1 & SP3

runid	S80313B1	S80317B1	S80317B2
Processor (DB server)	z14-16w	z14-16w	z14-16w
APP server O/S	SLES 12 SP3	SLES 12 SP3	SLES 12 SP3
App server config	4x LPAR	4x LPAR	4x LPAR
Processor (App servers)	128 IFL	128 IFL	128 IFL
SMT	no	no	no
Additional variables		SCHED_RR	RPS - All CPs
# of jobs	128 jobs	128 jobs	128 jobs
z/OS Utilization	55%	53%	57%
App server Utilization	73%	70%	75%
Total # of Accounts (60M)	60,000,000	60,000,000	60,000,000
Batch elasped time (sec)	7,635	7,964	7,388
Avg SAP App CPU time (sec)	5,515.80	5,394.14	5,467.40
Avg time spent in Db2 (sec)	588.94	663.51	577.84
Avg Db2 CPU Time (sec)	363.08	369.96	365.53
Avg DB2 client/server latency (usec)	291	317	263
ETR (Million accounts per hour)	28.29	27.12	29.24
ITR (DB server)	51.44	51.17	51.29
ITR (App server)	38.75	38.75	38.98
Avg Db2 Logging Rate (MB/sec)	329.58	317.20	342.42

Table 5: z14 Tuning Parameters @ 128 IFLs

8.0 References

1. Processor Unit Functions. IBM z14 Technical Guide (Section 3.5)

   http://www.redbooks.ibm.com/abstracts/sg248451.html?Open

2. IBM z Systems: Performance Report on Exploiting Large Memory for DB2 Buffer Pools with SAP

   http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102461

3. Large Systems Performance Reference for IBM Z

   https://www-304.ibm.com/servers/resourcelink/lib03060.nsf/pages/lsprindex/$file/SC28118721.pdf

4. z Systems Simultaneous Multithreading Revolution

   http://www.redbooks.ibm.com/abstracts/redp5144.html

5. IBM z Systems: Performance Report on Exploiting SMT for SAP Application Servers on z13

   http://www.redbooks.ibm.com/redpapers/pdfs/redp5287.pdf

6. SAP on IBM Z Reference Architecture: SAP for Banking

   https://www.sap.com/documents/2015/07/8aab7888-5b7c-0010-82c7-eda71af511fa.html