IBM MQ for z/OS on z14 Performance

Version 1.1 – March 2018

Tony Sharkey

IBM MQ Performance
IBM UK Laboratories
Hursley Park
Winchester
Hampshire

Notices

DISCLAIMERS
The performance data contained in this report was measured in a controlled environment. Results obtained in other environments may vary significantly.

You should not assume that the information contained in this report has been submitted to any formal testing by IBM.

Any use of this information and implementation of any of the techniques are the responsibility of the licensed user. Much depends upon the ability of the licensed user to evaluate the data and to project the results into their own operational environment.

WARRANTY AND LIABILITY EXCLUSION

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

In Germany and Austria, notwithstanding the above exclusions, IBM's warranty and liability are governed only by the respective terms applicable for Germany and Austria in the corresponding IBM program license agreement(s).

ERRORS AND OMISSIONS
The information set forth in this report could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; any such change will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time and without notice.

INTENDED AUDIENCE
This report is intended for architects, systems programmers, analysts and programmers wanting to understand the performance characteristics of *IBM MQ V9.0*. The information is not intended as the specification of any programming interface that is provided by IBM MQ. It is assumed that the reader is familiar with the concepts and operation of IBM MQ V9.0.

LOCAL AVAILABILITY
References in this report to IBM products or programs do not imply that IBM intends to make these available in all countries in which IBM operates. Consult your local IBM representative for information on the products and services currently available in your area.

ALTERNATIVE PRODUCTS AND SERVICES
Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

USE OF INFORMATION PROVIDED BY YOU
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

TRADEMARKS AND SERVICE MARKS
The following terms used in this publication are trademarks of their respective companies in the United States, other countries or both:

- **IBM Corporation:** IBM
- **Intel Corporation:** Intel, Xeon
- **Red Hat:** Red Hat, Red Hat Enterprise Linux

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

EXPORT REGULATIONS
You agree to comply with all applicable export and import laws and regulations.

Preface

In this paper, I will be looking at the improvements to our performance tests on MQ for z/OS as we moved from z13 to z14.

This paper is split into several parts:
Part one - Setting expectations of the hardware move.
Part two - General test performance and scalability.
Part three - MQ exploitation of cryptographic improvements.
Part four - Dataset encryption.
Part five - Storage Class Memory usage.
Part six - Using zEnterprise Data Compression with MQ archive logs.

Part one describes what may impact the expectations of moving workload from z13 to z14, and why it is not always straightforward.

Part two presents the results of measurements run first on z13 and then subsequently on z14. We also include scalability measurements to demonstrate how MQ performs when the number of processors is increased up to 32.

Part three looks at the performance benefits to components of MQ that utilize the cryptographic facilities offered on IBM Z.

Part four discusses where MQ is able to use dataset encryption and how the performance has been affected by z14.

Part five looks at the benefit of Storage Class Memory (SCM) moving from PCIe on zEC12 to Virtual Flash on z14 and where it might be appropriate to use SCM.

Part six discusses the benefits of using zEnterprise Data Compression (zEDC) to compress MQ archive logs.

Table of Contents

1 Setting expectations of the hardware move

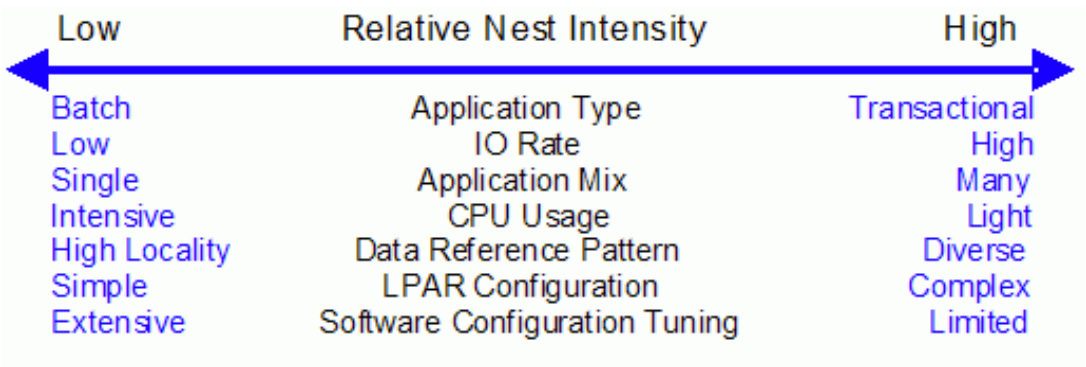
The IBM® z14™ (z14) offers many improvements over z13 but of particular note are both a higher processing speed and an increase in the number of processors available.

When trying to see what benefit you might get from moving to the z14, the Large System Performance Reference (LSPR) for IBM Z website is a good place to start. This documents a number of factors that may influence your particular workloads as well as providing a method to determine what improvement you may see.

It should be noted that when using the LSPR data to predict how a workload might perform on the z14, the type of workload makes a difference.

The most performance sensitive area of the memory hierarchy is the activity to the memory nest, namely the distribution of activity to the shared caches and memory.

Many factors influence the performance of a workload, however the Relative Nest Intensity (RNI) is typically the largest influencer.



Despite containing little business logic, the MQ performance workloads vary significantly in complexity and cover the entire range of Low, Average and High RNI.

Additionally, the number of processors allocated can affect the expectations – for example we have workloads that are classified as low RNI when running on 3 CP's but average when running on 32 CP's.

The following table shows the expected improvement on z14 over z13 for the varying workloads on the typical CPU configurations used in our performance tests:

CPUs	LOW	AVERAGE	HIGH
3	+6%	+8%	+11%
16	+7%	+10%	+12%
32	+17%	+11%	+14%

What this table suggests is that depending on the workload type and the number of CPUs allocated, we may see between 6 and 17% improvement over comparable measurements on z13.

As a rough guide we worked to the basic expectation of a 10% reduction in transaction cost when moving z13 to z14.

The performance sysplex used z13 used for performance measurements was a 4-drawer, whereas the z14 only had 2 drawers, which in some circumstances led to less optimal processor allocation to the LPARs.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

2 General test performance and scalability

General test performance

For the performance tests we typically saw performance in-line with the expected results from the LSPR tables, although there were some notable exceptions.

- Non-DataSharing shared queue workload using large messages offloaded to SMDS.

Whilst the intention was to compare environments as similar as possible, the z14 used was configured with 2 fewer CPC drawers than the z13. Subsequently some of the shared queue tests where all 3 LPARs were busy and the CPUs were allocated less than optimally, we did see reduced performance due to excessive cache miss.

- Cryptographic related workloads – channels protected using SSLCIPH specifications and queues protected using Advanced Message Security (AMS) policies.

The z14 introduced some significant improvements in the areas of cryptography – partly due to improvements on the CP Assist for Cryptographic Function (CPACF) feature which for MQ is largely used for encryption and decryption, but also with the introduction of the Crypto Express6S (CEX6S) feature.
For these measurements we were able to benefit from the CPACF improvements but deliberately chose to use the Crypto Express5S (CEX5S) feature, which is available to previous generation hardware. Performance information on the the CEX6S is detailed further in section "MQ exploitation of cryptographic improvements".

Our workloads using channels protected using cipher specification `ECDHE_RSA_AES_256_CBC_SHA384`, saw transaction cost reduce by up to 28% with a corresponding increase in throughput.

Queues protected by AMS policies saw a reduction in transaction cost of up to 35%.

- Channels using compression.

Both hardware compression "ZLIBFAST" and software compression "ZLIBHIGH" configurations saw a reduction in transaction cost of up to 20%.

- MQ dataset I/O performance.

Persistent private queue messaging tests saw up to a 15% increase in throughput.
Shared queue message message performance:

  - Up to 40% increase in throughput when backed by SMDS.
  - Up to 27% increase in throughput when backed by Db2 V12 Universal Table Space.

Scalability

For our scalability measurements we typically start with a non-persistent workload with the intent to be CPU limited rather than log constrained. The measurements use specific queues that are allocated to separate buffer pools and page sets for each particular workload to minimize any contention.

The workloads highlighted in this document are:

1. Non-persistent out-of-syncpoint using 2KB messages.
2. Non-persistent in-syncpoint using 2KB messages.

Basic configuration

Initially a pair of tasks are started, one requester and one server. These tasks use a pair of queues, one for the request message and one for the reply message. These queues are defined such that they use the same buffer pool and page set.

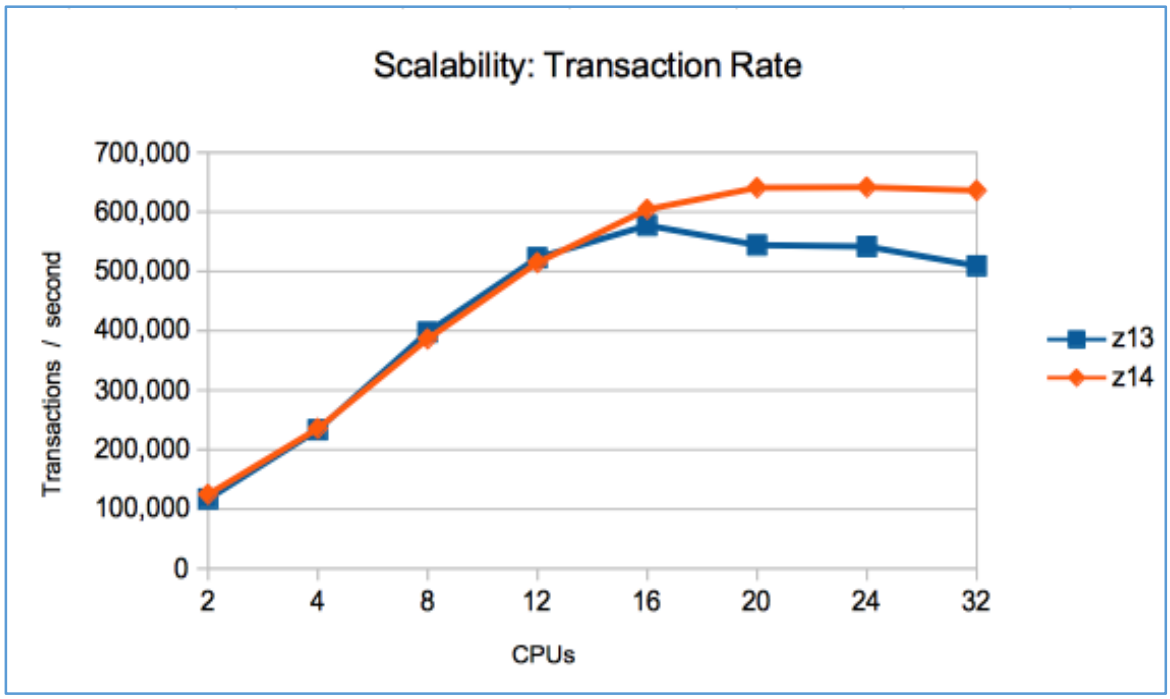
The requester puts a message and waits for a specific reply. When the requester gets that message, it generates a new request message and this continues until the applications are requested to end.

The server waits for a request message and upon successful get, generates a reply message and puts to the reply queue. When syncpoint is requested, the get and put will be performed within syncpoint.

The workload is increased with additional tasks that will use their own request and reply queues until there are 30 requesters, 30 servers and 30 pairs of queues. Each pair of queues is defined to a separate buffer pool and page set.

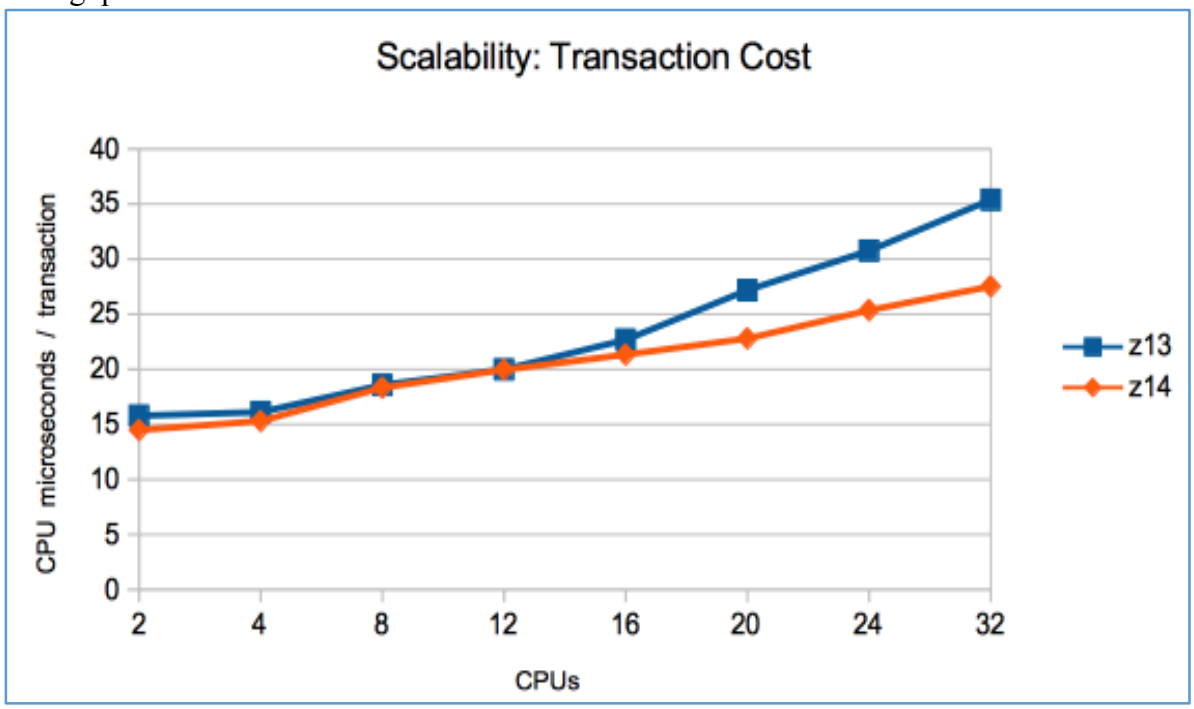
Non-persistent out-of-syncpoint using 2KB messages

The following chart shows the peak transaction rate achieved when the number of CPUs is increased.



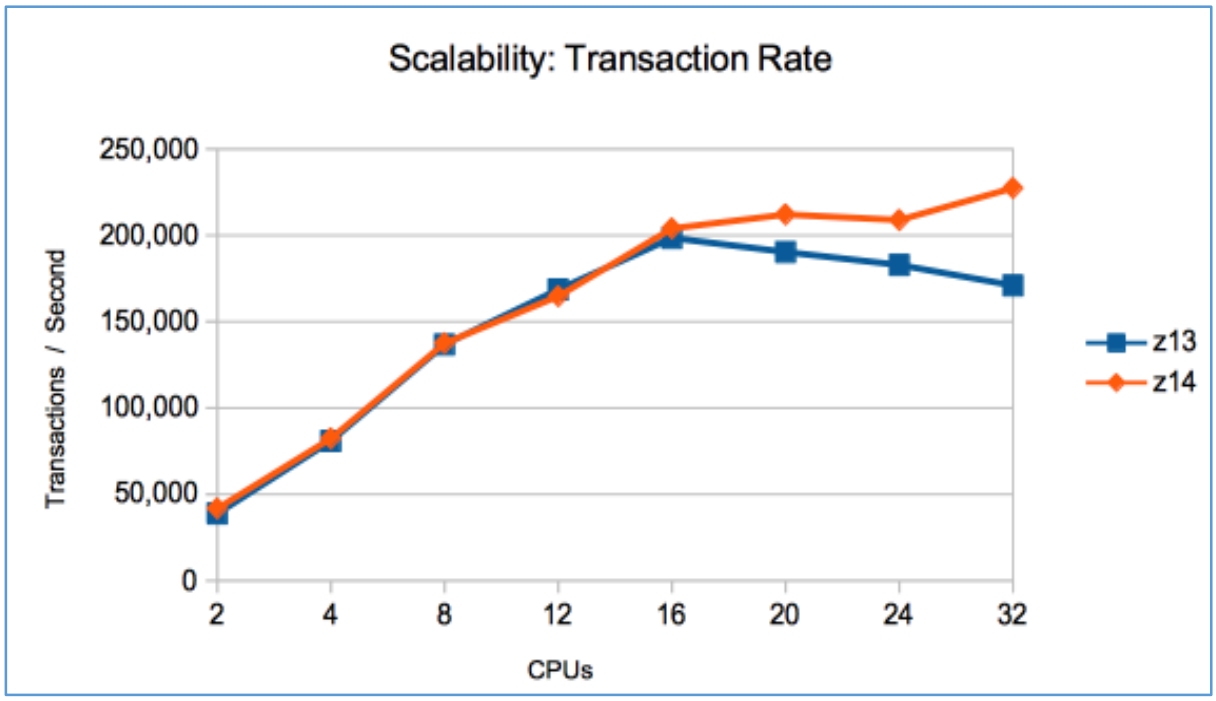
Where the z13 hit its peak throughput rate on a 16 CPU LPAR, the z14 continues to scale up to 20 CPUs, and does not see the rate tail off with additional CPUs. The absolute peak throughput is up 11% to 641,000 transactions/second, or 1.28 million messages/second.

The following chart shows the cost of a transaction when the workload is running at peak throughput:



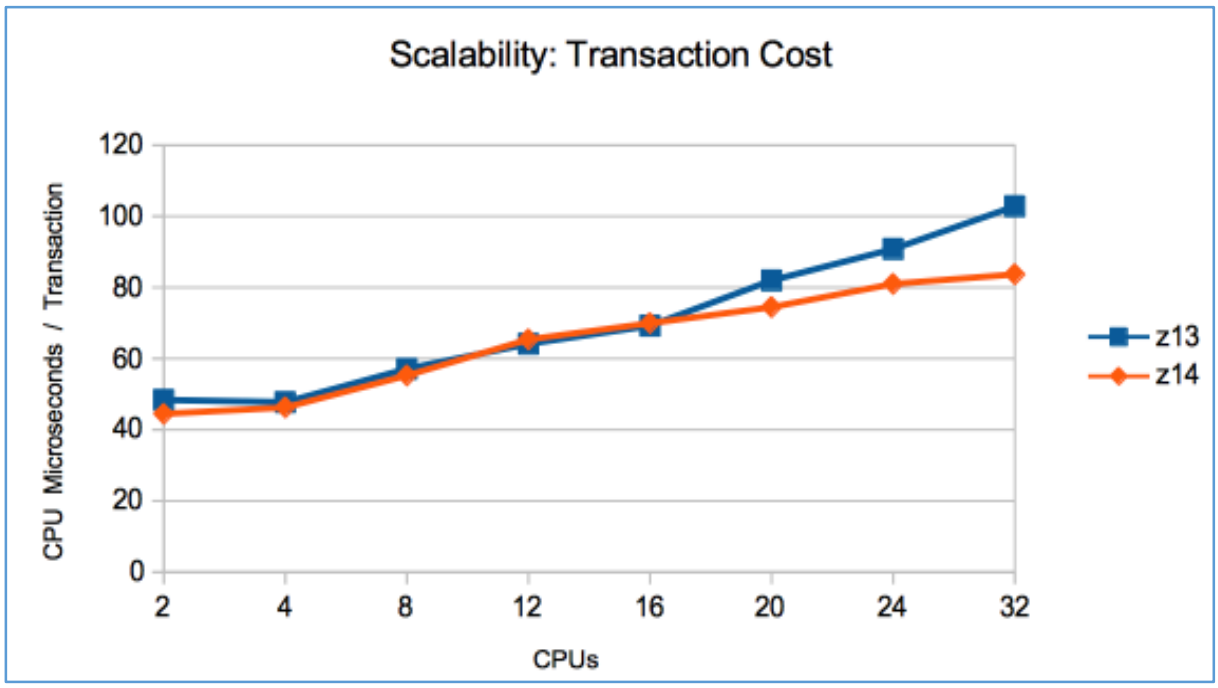
Non-persistent in-syncpoint using 2KB messages

The following chart shows the peak transaction rate achieved when the number of CPUs is increased.



Where the z13 hit its peak throughput rate on a 16 CPU LPAR and then dropped off with additional CPUs, the z14 continues to scale up to 32 CPUs. The absolute peak throughput is up 14% to 227,000 transactions/second.

The following chart shows the cost of a transaction when the workload is running at peak throughput:



3 MQ exploitation of cryptographic improvements

As mentioned in the previous section, the z14 benefits from some significant improvements in the cryptographic area – both Crypto Express 6S (CEX6S) and CPACF. Further information relating directly to the cryptographic performance is available in the document titled "IBM z14 Cryptographic Performance".

This section details the performance improvements observed in our MQ performance tests for the following classes of tests:

- Channels protected with SSLCIPH specifications.
- Queues protected using AMS policies.

Information on dataset encryption is reported separately in section "Dataset Encryption".

The comparisons are between:

- IBM z13 with Crypto Express5S (CEX5S)
- IBM z14 with Crypto Express5S
- IBM z14 with Crypto Express6S (CEX6S)

Channels protected using SSLCIPH specifications

When performance testing channels protected with cipher specifications, we currently use the following 3 ciphers:

1. `ECDHE_RSA_AES_256_CBC_SHA384`
2. `TLS_RSA_WITH_AES_256_CBC_SHA256`
3. `ECDHE_ECDSA_AES_256_CBC_SHA384`

In terms of performance, the `TLS_RSA_WITH_AES_256_CBC_SHA256` cipher is considerably lower cost at the time of secret key negotiation. For example, when we renegotiate the secret key every 1MB, the `TLS_RSA_WITH_AES_256_CBC_SHA256` cipher is approximately 25-40% lower cost than the `ECDHE` prefixed ciphers.

Outside of key negotiation, the 3 cipher specifications deliver comparable performance at similar cost.

For the purposes of this section, we will compare performance results using only cipher specification `ECDHE_RSA_AES_256_CBC_SHA384`, with non-persistent messages of 32KB. The measurements use a request/reply workload between 2 queue managers on separate LPARs, that are connected by a 10Gb low-latency link.

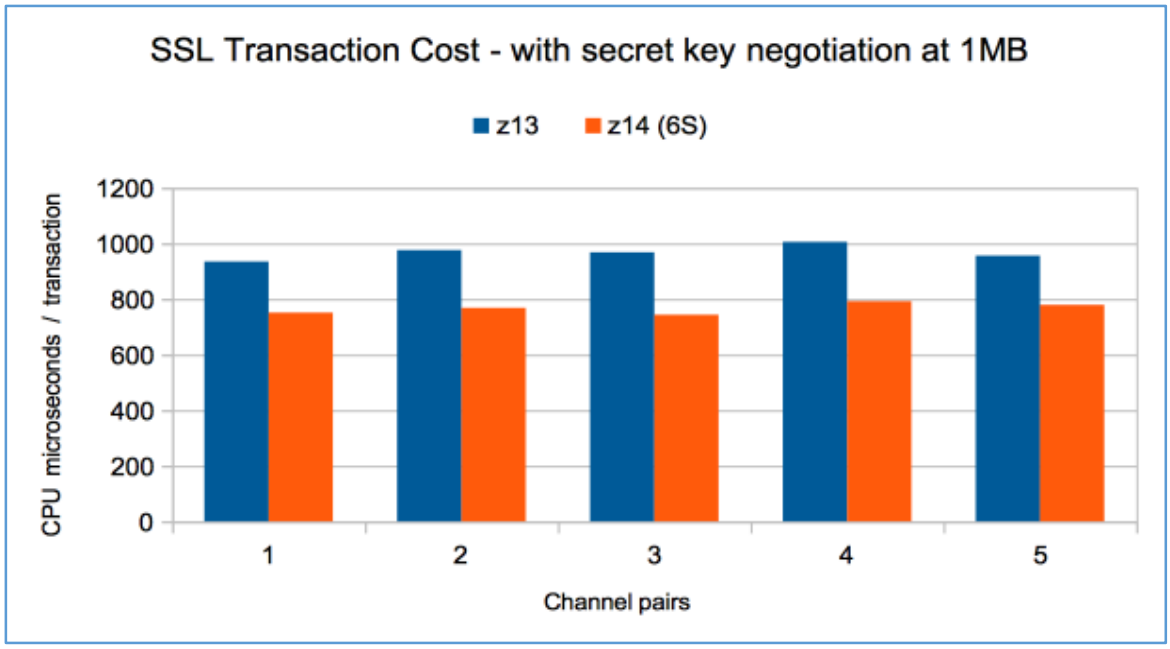
Measurements are run in 2 configurations:

1. Negotiate the secret key every 1MB that passes over the channel, by setting SSLRKEYC(1048576).
    a. Demonstrates the benefits of CEX6S and CPACF.
2. Negotiate the secret key at channel start only – with the channels remaining active for the duration of the workload.
    a. Demonstrates the benefits of CPACF only.

Negotiate the secret key every 1MB

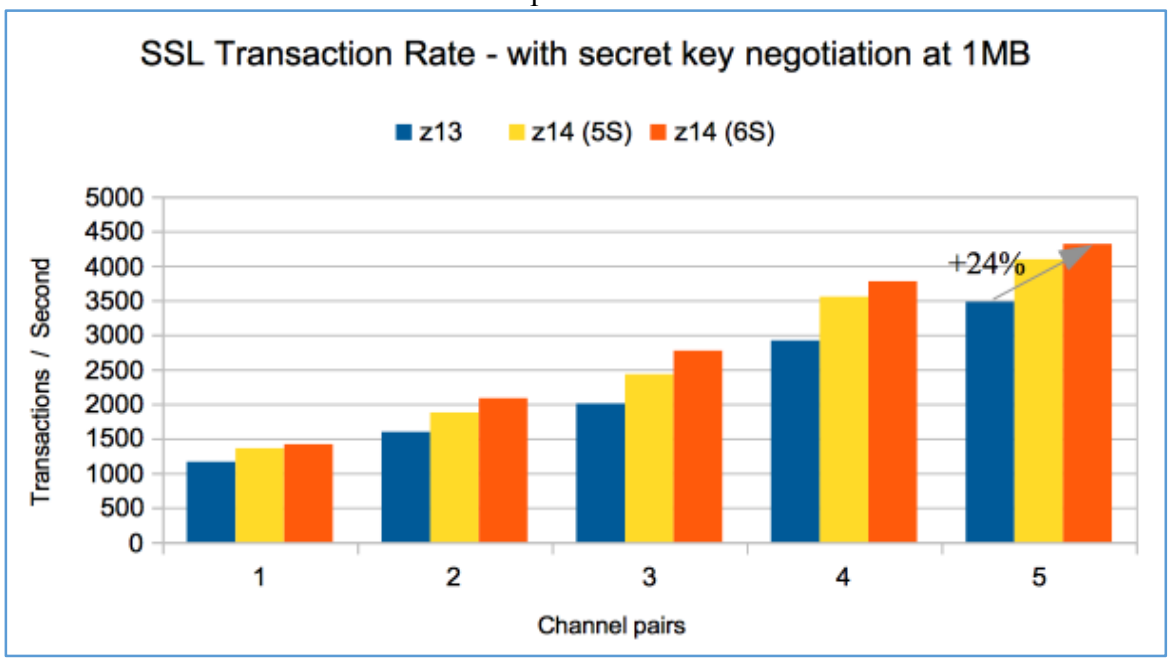
When negotiating the secret key, MQ is able to offload a significant proportion of the processing to the Crypto Express feature. For data encryption, the encryption and decryption is processed by CPACF, which remains constant on the z14.

The following chart compares the cost of the workload between z13 and z14. Note that the z14 transaction costs are not affected by the level of Crypto Express, so only data from the CEX6S measurement is shown.



The transaction cost chart shows that in our measurements, the cost was on average 20% lower on z14 than the equivalent measurement on z13.

In the second chart, the transaction rate achieved by the measurements is shown. In this instance we see evidence of the lower response time from the CEX6S.

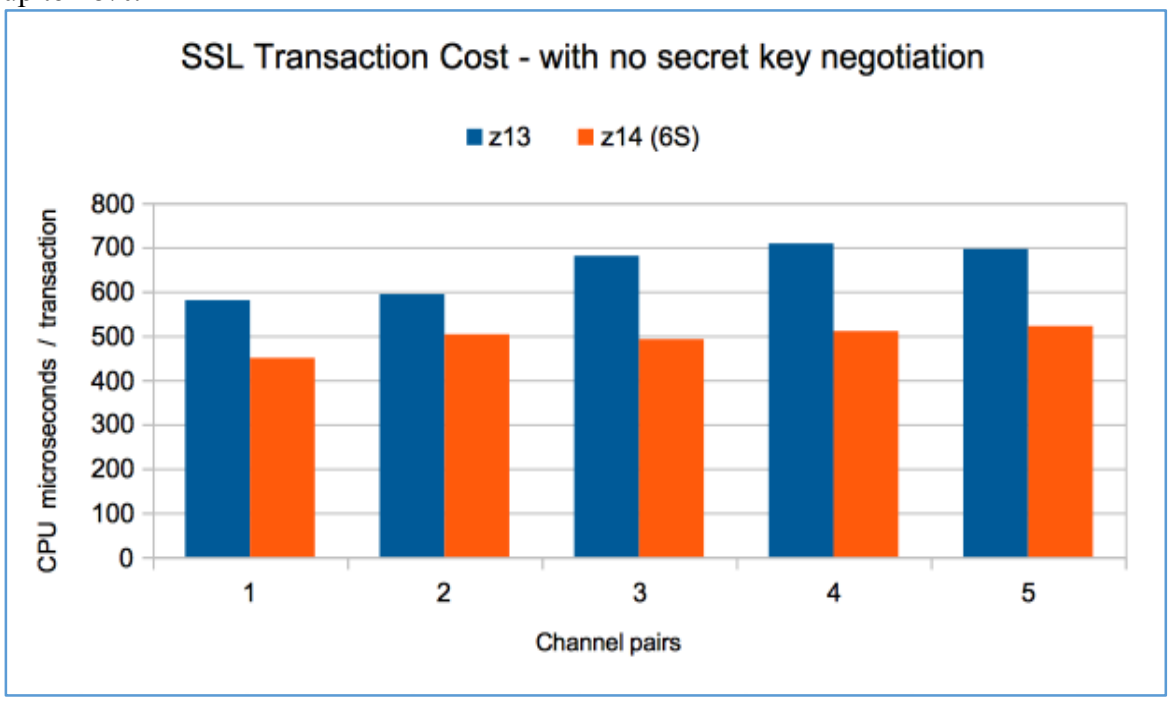


Across the workload, the improvements in throughput between z13 and z14 (CEX5S) was 17-22%. The CEX6S increased the throughput a further 5-14%.

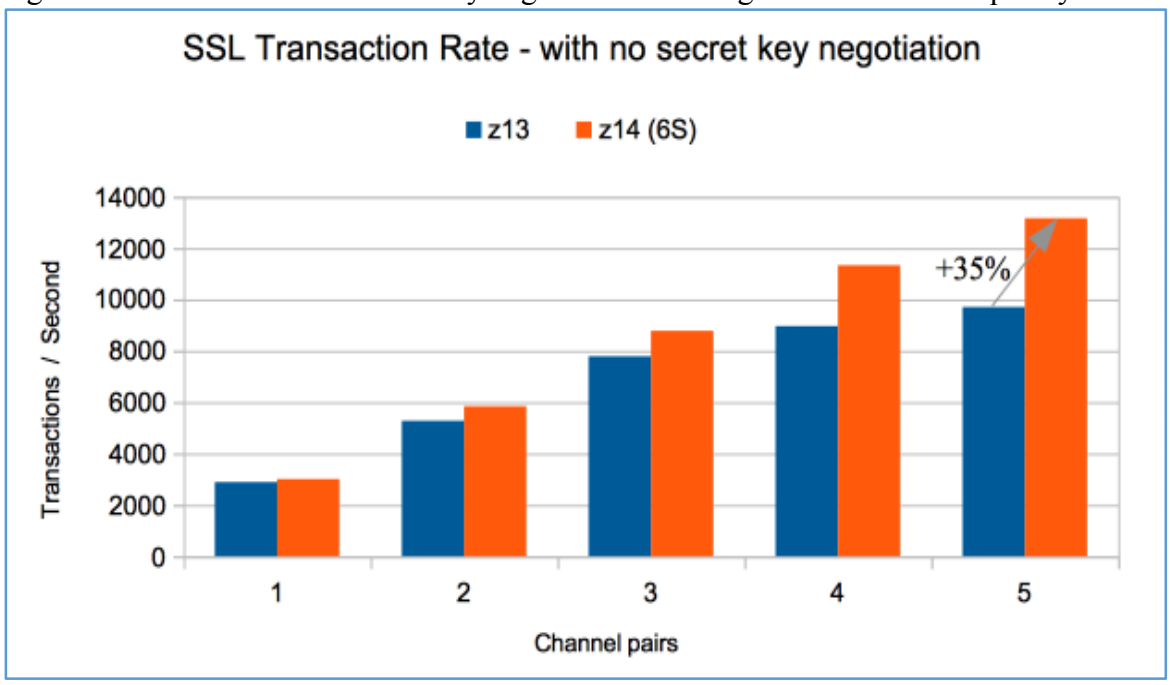
Negotiate the secret key only at channel start

By negotiating the secret key only at channel start, these measurements are aimed at demonstrating the improved performance of encryption/decryption services.

The following chart compares the cost of the workload between z13 and z14. As there is no secret key negotiation involved in the measurement, the level of Crypto Express is irrelevant for the purposes of this measurement. The chart demonstrates that z14 is considerably lower cost for these types of workloads than z13 – our tests show a reduction in transaction cost of up to 28%.



The second chart shows the transaction rate achieved when the secret key is not negotiated on a regular frequency. The transaction rate is up to 35% higher on z14 and is significantly higher than achieved when secret key negotiation is configured at a 1MB frequency.



Queues protected using AMS policies

When comparing the performance of queues protected with AMS policies we used a simple request/reply model using small (2KB), medium (64KB) and large (4MB) messages.

The policies were applied to both the request and reply queues where:

- Integrity used messages signed with SHA256.
- Privacy used message signed with SHA256 and encrypted using AES256.
- Confidential used messages encrypted using AES256 and the key reused 32 times.

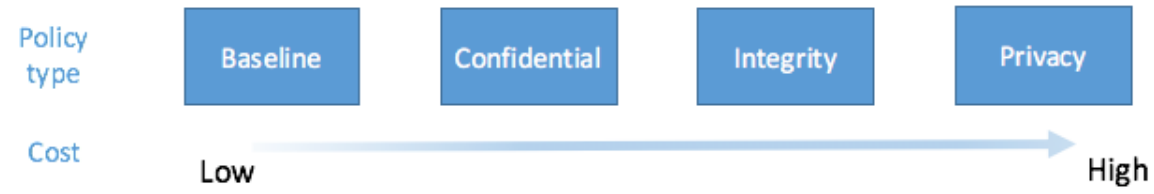
Significant performance improvements to AMS protection were applied to IBM MQ on z/OS in Continuous Delivery Release version 9.0.1 as documented in the performance report.

AMS Integrity is largely impacted by the response times of the Crypto Express feature. In terms of cryptographic hardware usage, message size does not impact the cost protecting the message using Integrity.

AMS Privacy performance is impacted both by the response time of the Crypto Express feature and the performance of the CPACF which is used to encrypt and decrypt the message.

AMS Confidential performance is largely impacted by the performance of the CPACF encryption and decryption of the message. Message size is a large factor in the improvements from running on z14.

In basic terms, the costs of AMS protection can be considered thus:



AMS – Small messages

The following table show the percentage difference between z13 with CEX5S and z14 with CEX6S.

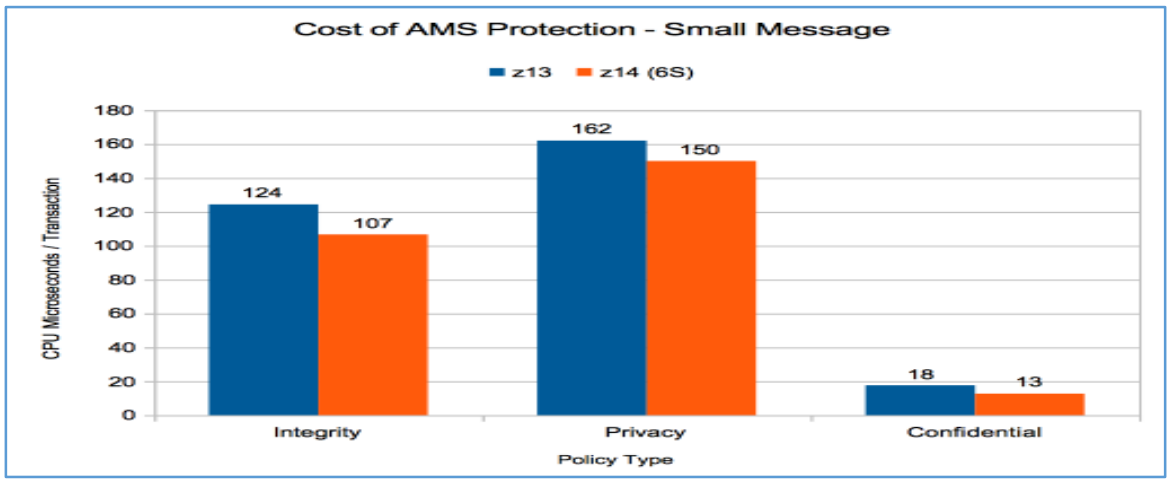
	Integrity	Privacy	Confidential
% change in transaction cost	-13	-7.4	**-15.4**
% change in cost of protecting message with AMS	-14	-7.5	**-28**
% increase in throughput	+29	+20	+12.4

The '% change in cost of protecting message with AMS' is based on a comparison of the workload being run both with and without the AMS policy being defined on the queues. The following table demonstrates how this number was derived. Costs shown are in CPU microseconds per transaction.

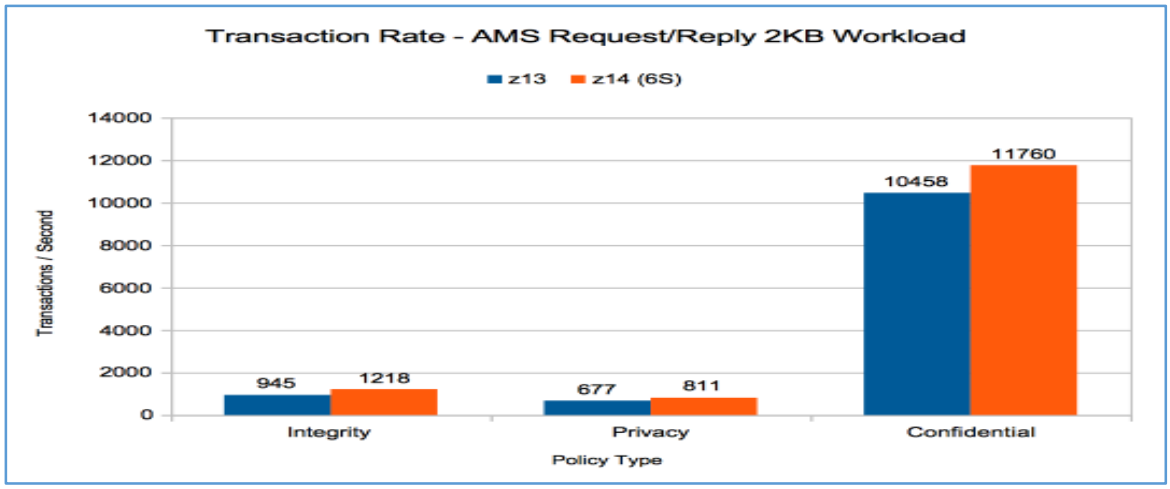
	z13	z14	% change
Un-protected cost	52.87	49.15	-7
AMS Confidential cost	88.18	74.56	**-15.4**
Cost of protection Calculated: AMS cost – un-protected cost	35.31	25.41	**-28**

The following chart shows the cost of AMS protection for both z13 and z14. *This cost does not include the cost of the basic MQPUT or MQGET as reported in the class 3 accounting data.*

Furthermore, it is the cost of 1 message being protected and subsequently un-protected. This means that the impact of protecting the messages in a transaction is double the values shown in the chart.



The second chart shows the achieved transaction rate for these workloads.



For small messages, the largest benefit is for messages protected using the Integrity level of protection.

AMS – Medium messages

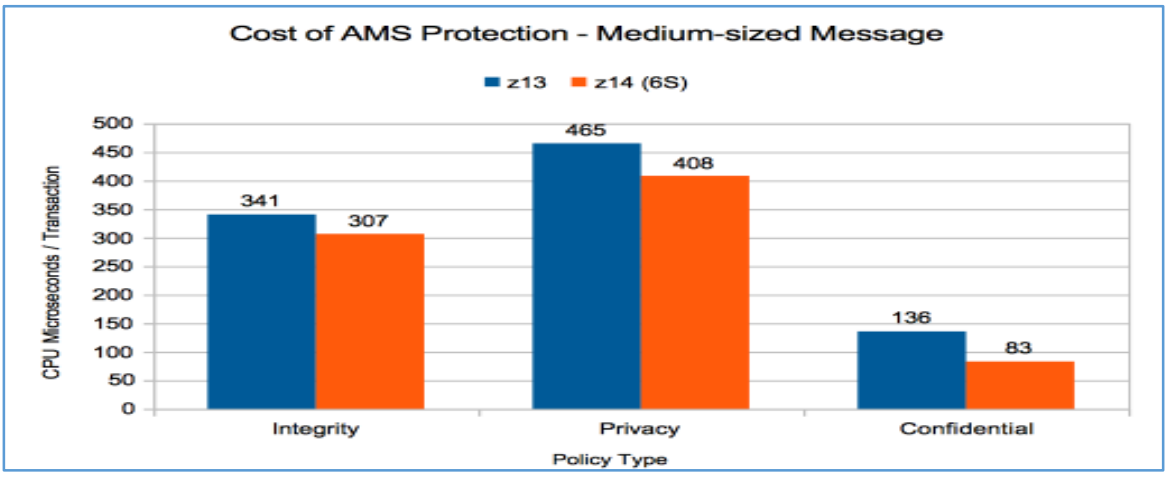
The following table show the percentage difference between z13 with CEX5S and z14 with CEX6S for medium-sized messages.

	Integrity	Privacy	Confidential
% change in transaction cost	-10	-12	-30
% change in cost of protecting message with AMS	-10	-12.25	-40
% increase in throughput	+21.3	+18.5	+31.3

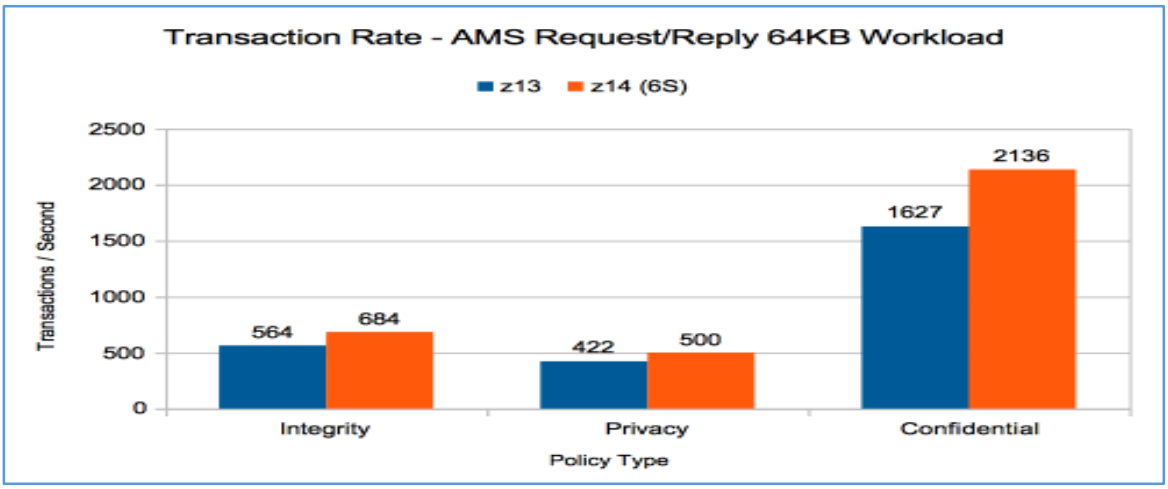
With medium size messages, the impact of the CEX6S is reduced, but this is offset for AMS protection that involves encrypting the message by improvements to CPACF.

The following chart shows the cost of AMS protection for both z13 and z14. *This cost does not include the cost of the basic MQPUT or MQGET as reported in the class 3 accounting data.*

Furthermore, it is the cost of 1 message being protected and subsequently un-protected. This means that the impact of protecting the messages in a transaction is double the values shown in the chart.



The second chart shows the achieved transaction rate for these workloads.



AMS – Large messages

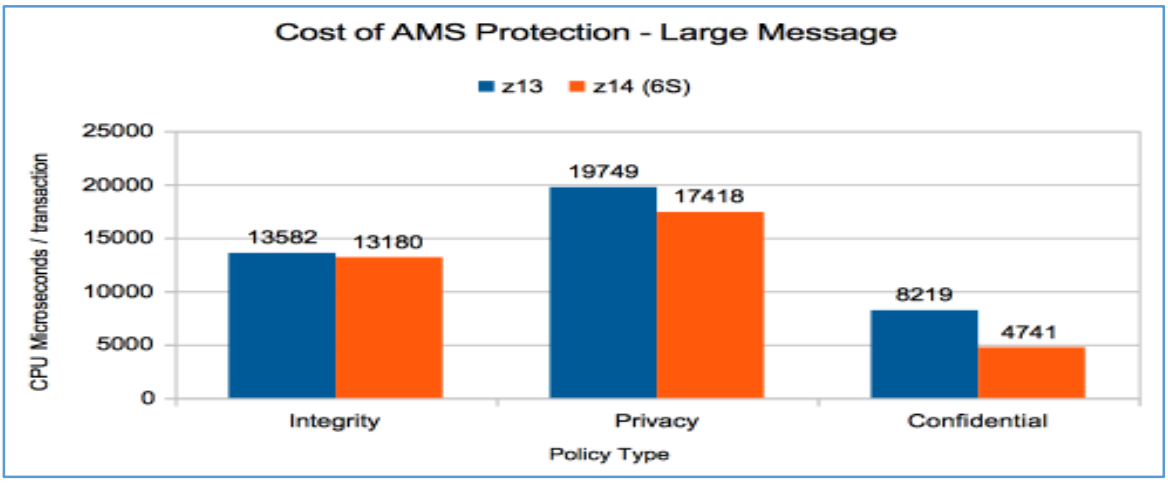
The following table show the percentage difference between z13 with CEX5S and z14 with CEX6S for large messages.

	Integrity	Privacy	Confidential
% change in transaction cost	-4.6	-12	-36
% change in cost of protecting message with AMS	-3	-12	-42.2
% increase in throughput	+9.5	+13	+36

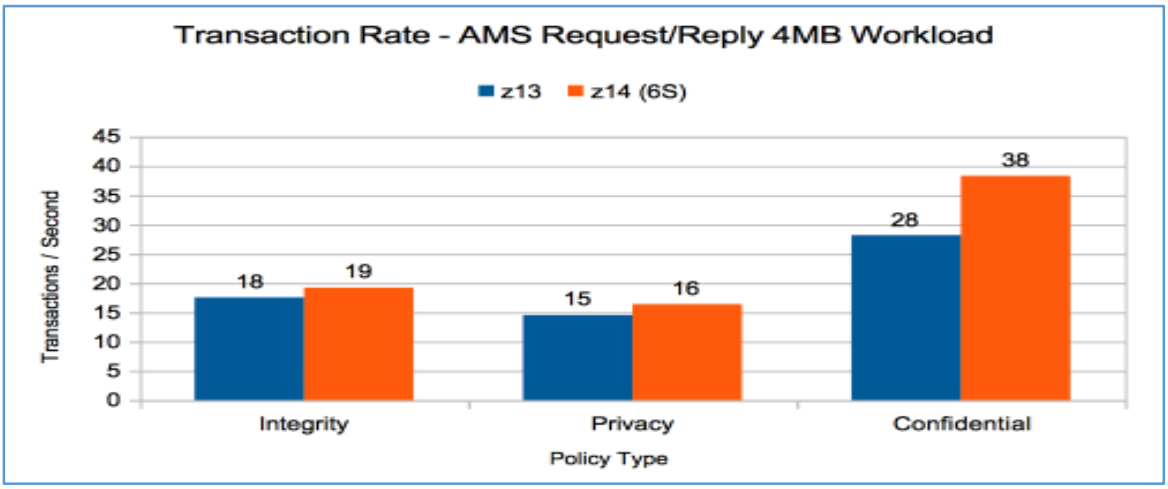
With large messages, the performance gains are largely coming from CPACF (encryption / decryption) and general z14 CPU improvements.

The following chart shows the cost of AMS protection for both z13 and z14. *This cost does not include the cost of the basic MQPUT or MQGET as reported in the class 3 accounting data.*

Furthermore, it is the cost of 1 message being protected and subsequently un-protected. This means that the impact of protecting the messages in a transaction is double the values shown in the chart.



The second chart shows the achieved transaction rate for these workloads.



4 Data set encryption

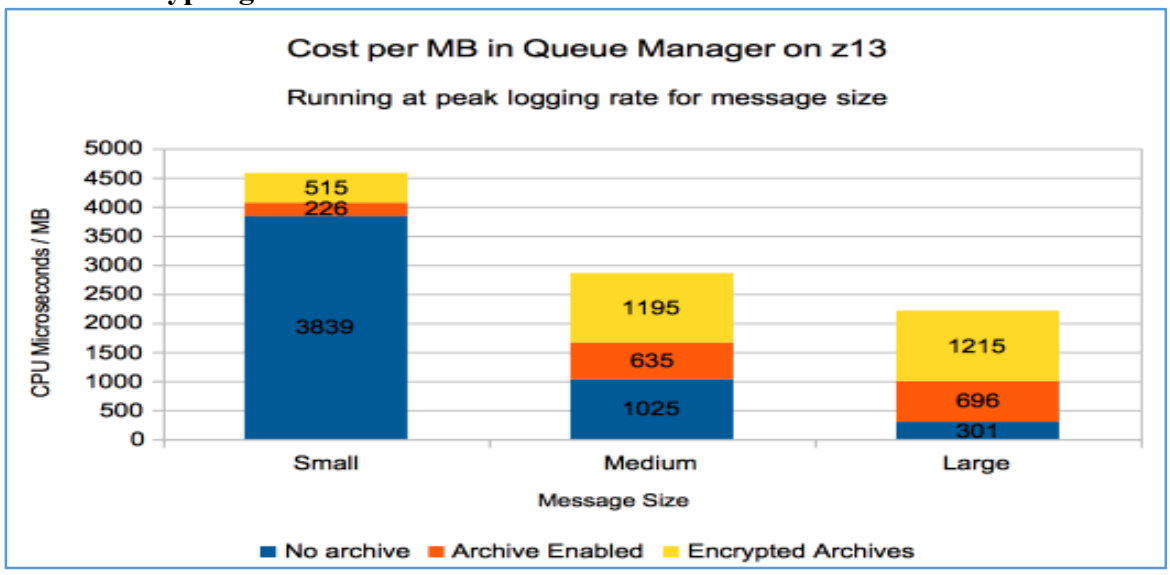
Data set encryption (DSE) became available for z/OS v2r2 via APAR OA50569.

MQ supports DSE only for archive and BSDS, with AMS being the preferred mechanism for encrypting messages at rest.

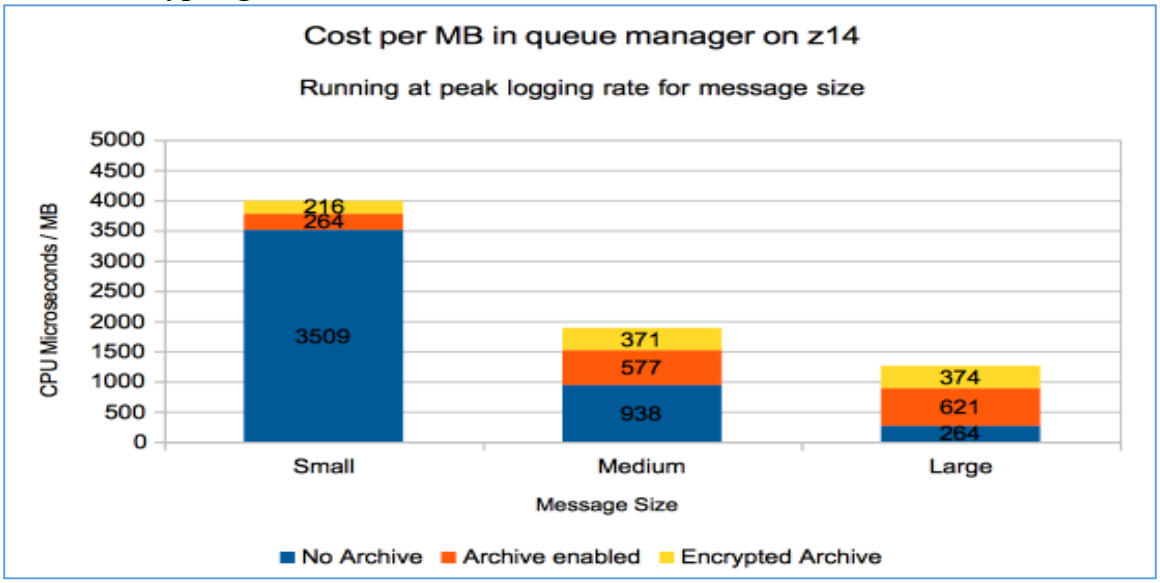
In the blog "MQ and the use of data set encryption for IBM z/OS v2.2" we suggest that the cost of encrypting archives on z13 is approximately 3 times the cost of archiving.

Using the same methodology on the z14 as detailed in the blog, the cost of encrypting archives is approximately 1.6-1.8 times the cost of archiving.

Cost of encrypting archives on z13:



Cost of encrypting archives on z14:



5 Storage Class Memory usage

The first implementation of Storage Class Memory (SCM) was on zEC12 where it was hosted on Flash Express (SSD on PCIe cards). This allowed the SCM to be allocated to the Coupling Facility to provide a greater amount of CF storage at reduced cost.

With the z14, SCM uses Virtual Flash Memory (VFM) as a direct replacement for Flash Express, and VFM provides much simpler management and better performance by eliminating the I/O adapters located in the PCIe drawers.

The IBM MQ V8 Features and Enhancements redbook discusses the use of SCM with IBM MQ and the use cases where it may be beneficial.

There are 2 primary use cases:

1. Emergency storage
   SMDS and message offloading can be used in conjunction with SCM to reduce the likelihood of an MQRC_STORAGE_MEDIUM_FULL reason code being returned to an IBM MQ application during an extended outage.

2. Improved performance
   Using SCM to increase the number of messages that can be stored on a shared queue without incurring the performance cost of using SMDS.

There are configurations where SCM will not benefit MQ application structures:

1. Non-sequential access of MQ messages
   - SCM uses the KEYPRIORITY1 algorithm with MQ shared queues to determine the order that messages are written to SCM (pre-staging) and the order messages are migrated back into the CF (pre-fetching).
   - Both pre-staging and pre-fetching are typically performed asynchronously to reduce the likelihood of the application being blocked which synchronous I/O to/from SCM occurs.
   - Pre-fetching using the KEYPRIORITY1 algorithm works on the assumption that the messages will be gotten in MQ message priority order. Multiple messages are pre-fetched with the number dependent on the message size.
   - When processing messages out of priority order, the pre-staging and pre-fetching functions controlled by the KEYPRIORITY1 algorithm is unable to accurately predict which messages can be pre-staged and which messages need to be pre-fetched next.
     - This can result in significantly more expensive MQPUTs and MQGETs.
     - Example costs are documented in performance report MP16 in section “Non-Sequential gets from deep shared queue”.

When the pre-fetch is working efficiently, the benefits are minimal when compared to z13. Moving the SCM to Virtual Flash does mean that the pre-fetch response time is reduced, meaning that there is less risk of SCM delays occurring.

To demonstrate the impact of moving SCM from Flash Express to Virtual Flash Memory, we use the use case which is sub-optimal for MQ, namely non-sequential access of MQ messages.

In this example, we pre-loaded a shared queue with 5 million messages.
The application randomly selects messages to be got/put using a known CORRELID.

	z13	z14	Change
Transaction rate	483	804	+66%
RMF data			
Read Average Service Time	1285.4	322.6	-74.9%
Write Average Service Time	414.1	230.7	-44.3%
SCM Read Count	39,000 / minute	77,000 / minute	+41%
SCM Delayed	25%	26.2%	

In each case the percentage of requests delayed due to the required message not being in CF and having to be synchronously pre-fetched from SCM is similar on z13 and z14.

Despite this, the z14 transaction rate is 66% higher, which is largely due to the reduced time to fetch the data from SCM to CF.

6 Using zEDC to compress MQ archive logs

MQ offers the use of zEnterprise Data Compression (zEDC) for both channel compression and archive log compression.

The performance of channel compression is discussed briefly in the "General test performance" section.

Previously we have blogged about using zEDC with MQ archive logs in terms of potentially reducing the storage requirements for archive logs, but this section provides an update discussing the benefits of z14.

For these measurements we have re-used the process and configurations detailed in the blog, i.e. we are using an MQ queue manager configured with dual logs and dual archives. The measurements are configured to use messages that vary in compressibility from 0 to 80%.

One of the most notable benefits of using zEDC on MQ archive logs on the z13 performance system was to reduce the load on the disk subsystems' cache such that the MQ log performance was improved by up to 45% in terms of log write rate. This improvement did come at a cost to CPU and an increase in recovery time.

z13 performance

In the original blog post, we calculated the increase in QM TCB time for the zEDC workload, which was based on the total CPU cost of the workload. Given that in the high throughput workloads, typically larger messages, the reduced load on the IO subsystem resulted in higher throughput, the costs reported were calculated on total data processed rather than costs per MB.

Revisiting these values and calculating based on MB of log data processed gives the following table of values:

Message Size	4KB	32KB	1MB	4MB
Increase in QM TCB over non-zEDC measurement	+7%	+10-30%	+40-70%	+50-75%
Increase in peak log throughput	0%	0%	+15-30%	+20-45%

Whilst the log and archive write rates are important, in event of failure, the restart/recovery time is also important.

When recovering 4GB of data, using a range of message sizes from 64KB to 4MB, the following performance observations were made:

	Uncompressed Archives	Archives compressed using zEDC
Recovery rate (MB/sec)	88	27
Cost per MB (CPU microsecond)	860	1900

- Recovery rate of the compressed archives is 30% of the uncompressed archives.
- Recovery cost per MB of archive logs is 2.2 times that of uncompressed archives.

z14 performance

The impact of compressing MQ archive logs on z14 is more notable for a number of reasons;

1. The increase to the log rate when compressed archives is enabled is more significant.
2. The cost of compressing archive log is reduced.
3. The cost of recovery of data from compressed archives is reduced from z13.
4. The rate of recovery of data from compressed archives is increased, compared to z13.

In terms of the costs of compressed archives:

Message Size	4KB	32KB	1MB	4MB
Increase in QM TCB over non-zEDC measurement	Up to 4%	7%	15-25%	7-14%
Increase in peak log throughput	0%	Up to 4%	44-72%	50-94%

In the worst case, the cost of archiving increase by 25%, which was for an incompressible message payload of 1MB messages.

The side effect of highly compressible messages resulting in lower load on the I/O subsystem meant that the impact to the peak log rate increased from 272MB/sec to 530MB/sec per log, when achieving 80% compression on 4MB messages.

Recovery of archive logs saw improved performance on z14 too:

	Uncompressed Archives	Archives compressed using zEDC
Recovery rate (MB/sec)	110	38
Cost per MB (CPU microsecond)	740	1148

- Recovery rate of the compressed archives is 35% of the uncompressed archives.
- Recovery cost per MB of archive logs is 1.5 times that of uncompressed archives.
    - Recovery rate using compressed archives is 40% faster than on z13.

How many zEDC features were used?

In these measurements there was 1 zEDC feature in use. At heaviest usage, it was 70% utilised based on RMF's PCIE Activity Report.

Appendix A – Test Environment

Measurements were performed using:

The IBM MQ performance sysplex ran measurements on:

- **IBM z13 (2964-7E1) – 4 CPC drawers**
- **IBM z14 (3906-761) – 2 CPC drawers**

The sysplex was configured thus:

- LPAR 1:
  - 2-32 dedicated general purpose processors with 128 GB of real storage.
- LPAR 2:
  - 3-10 dedicated general purpose processors with 32 GB of real storage.
- LPAR 3:
  - 3 dedicated general purpose processors with 32 GB of real storage.
- z/OS v2r2.
- Db2 for z/OS version 12 configured for MQ using Universal Table spaces.
- MQ queue managers:
  - configured at MQ V9.0.3.
  - configured with dual logs and dual archives.

Coupling Facility:

- Internal Coupling Facility with 4 dedicated processors
- Coupling Facility running latest CFCC level.
- Dynamic CF dispatching off
- 3 x ICP links between z/OS LPAR and CF.

DASD:

- FICON Express 16S connected DS8870
- 4 dedicated channel paths
- HYPERPAV enabled
- zHPF disabled for the purposes of the testing

Network:

- 10GbE network configured with minimal hops to distributed machine
- 1GbE network available