IBM MQ for z/OS on z15 Performance

Version 1.0 – January 2020

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 V9.1*. 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 z14 to z15.

This paper is split into several parts:
Part one - Setting expectations of the hardware move.
Part two - General test performance and scalability.
Part three - Channel Compression use of On-Chip compression accelerator.
Part four - Archive log compression using On-chip compression accelerator.
Part five - Improvements from Crypto Express 7S

Part one describes what may impact the expectations of moving workload from z14 to z15, and why it is not always straightforward.

Part two presents the results of measurements run first on z14 and then subsequently on z15. 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 MQ channels when compression is applied, with particular benefit from the on-chip compression accelerator that replaces the IBM zEnterprise Data Compression (zEDC) Express feature.

Part four discusses the benefits of using on-chip compression accelerator to compress MQ archive logs.

Part five looks at the performance benefits to components of MQ that utilize the cryptographic facilities offered on IBM Z, in particular those using Cryptographic co-processor and accelerator function.

Table of Contents

1 Setting expectations of the hardware move

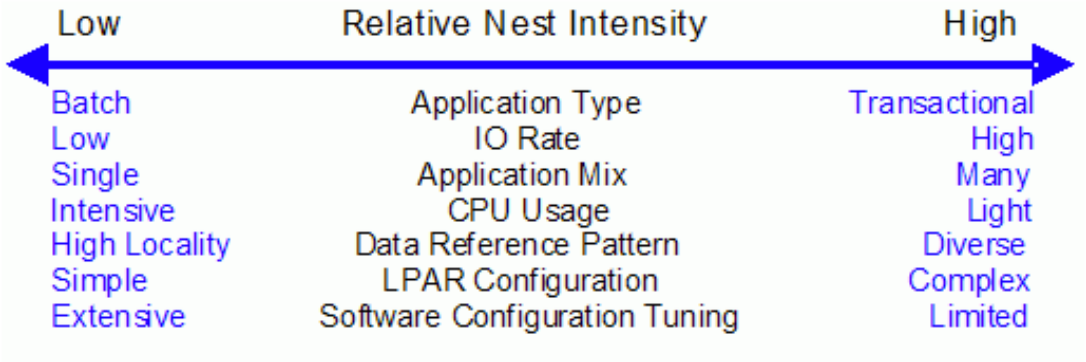
The IBM® z15™ (z15) offers many improvements over z14 but of particular note are an increase in the number of processors available, increased L3 cache size and the on-chip compression accelerator "IBM Integrated Accelerator for zEnterprise Data Compression" which can be used for both MQ channel compression and MQ archive log compression.

When trying to see what benefit you might get from moving to the z15, 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 z15, 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 z15 over z14 for the varying workloads on the typical CPU configurations used in our performance tests:

CPUs	LOW	AVERAGE	HIGH
3	+11%	+13%	+15%
16	+12%	+14%	+16%
32	+12%	+13%	+16%

What this table suggests is that depending on the workload type and the number of CPUs allocated, we may see between 11 and 16% improvement over comparable measurements on z14.

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.

- Scalability tests achieving up to 19% higher throughput.
  Non-persistent out-of-syncpoint workload achieved in excess of 1.5 million messages per second on single z/OS LPAR with 20 CPUs.
  Non-persistent in-syncpoint workload achieved in excess of 0.5 million messages per second on a single z/OS LPAR with 20 CPUs.

- Channels using compression.
  Hardware compression using COMPMSG(ZLIBFAST) saw a reduction in transaction cost of up to 42% and increase in throughput of up to 90%.

- Channels protected with TLS ciphers (SSLCIPH).
  On simple request/reply workloads we saw channel throughput increase up to 53%. Performance gains are dependent on the cipher used to protect the workload and the frequency of secret key negotiation.

- Queues protected with AMS policies.
  Transaction costs were up to 12% lower on z15, with a corresponding increase in transaction rate.

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.

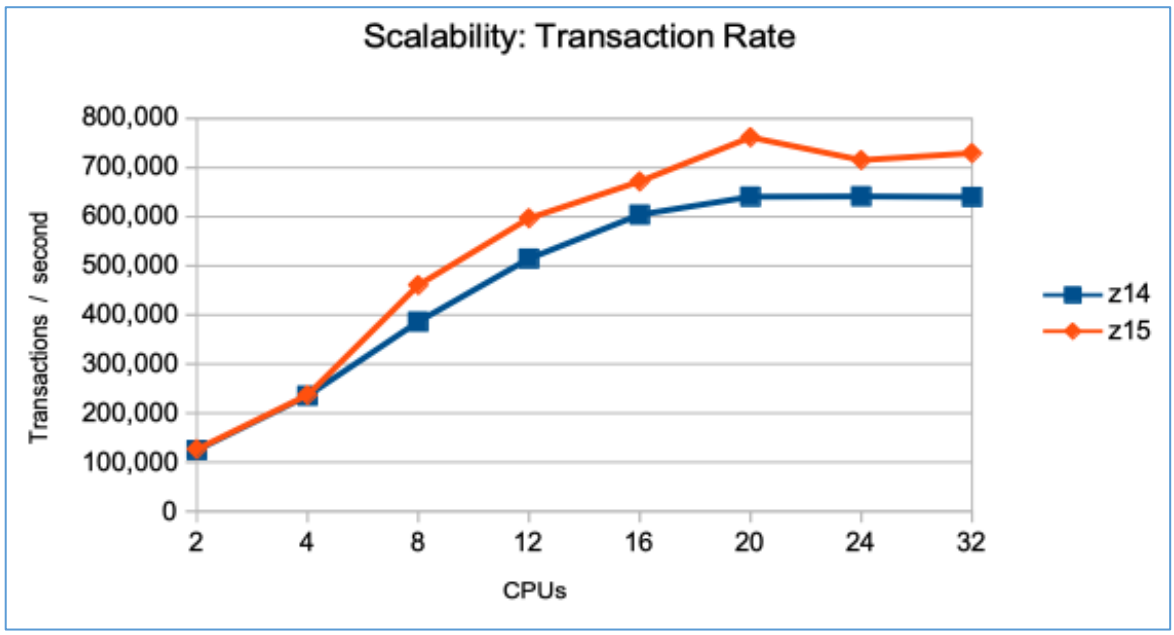
The requester puts a message and waits for a specific reply. When the requester gets that message, it generates a new request message and this continues until the applications are requested to end.

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.

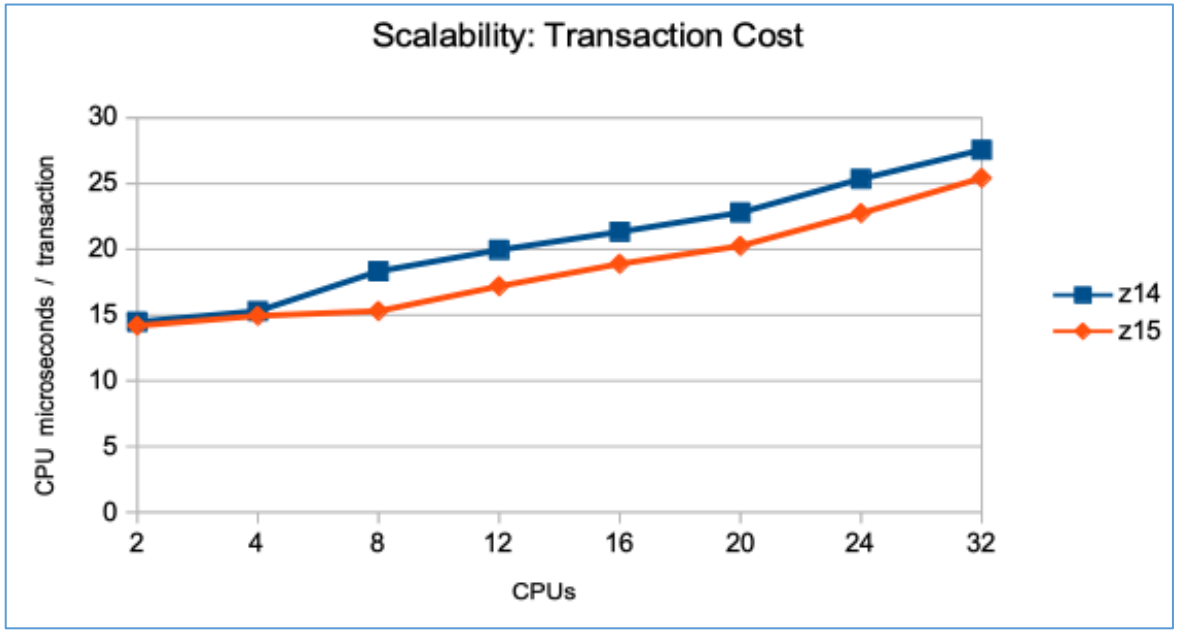
Non-persistent out-of-syncpoint using 2KB messages

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



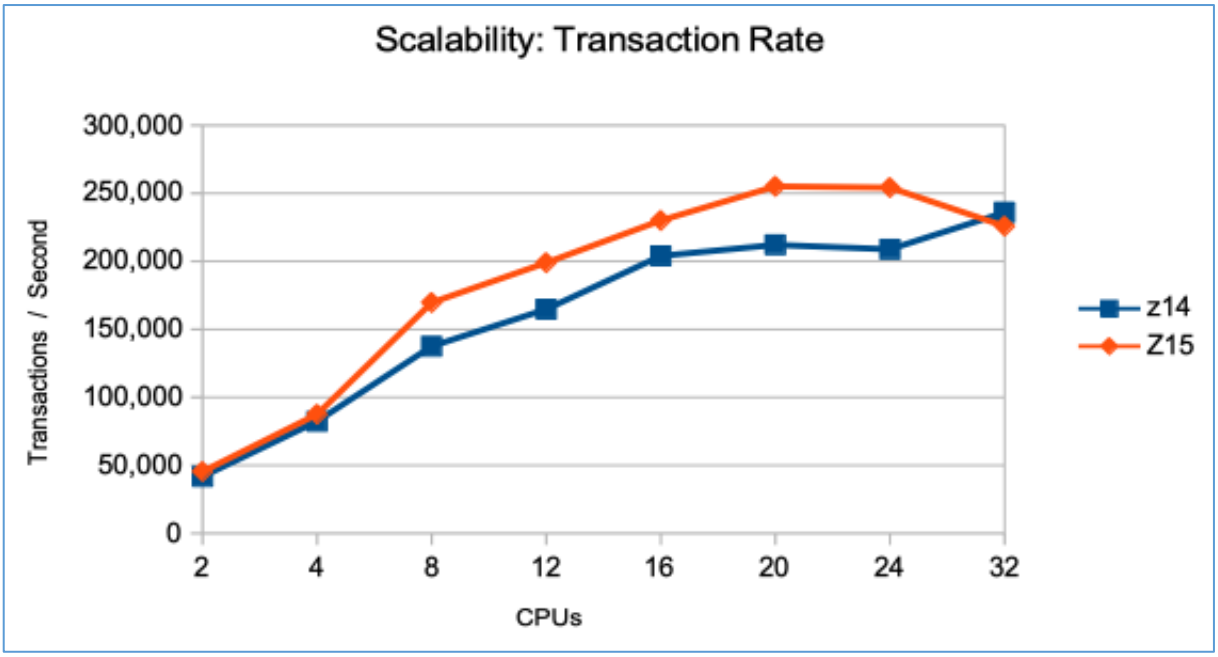
On both z14 and z15, the peak throughput for a workload on a single MQ queue manager was achieved with 20 CPUs, however the z15's throughput was 19% higher and achieved 761,000 transactions/second, or 1.52 million messages per second.

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



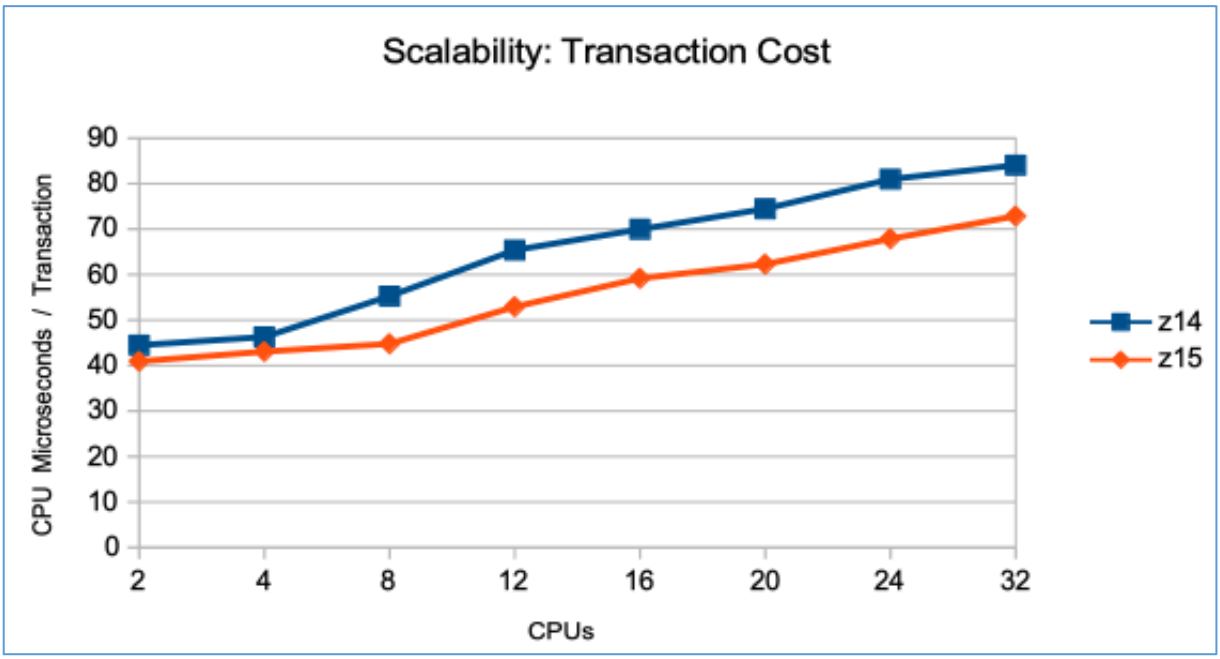
Non-persistent in-syncpoint using 2KB messages

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



The overall peak throughput achieved increases 12% on the z14 performance, but whereas the peak rate on z14 required 32 processors, the z15 was able to achieve its peak with 20 CPUs. The absolute peak throughput is up to 255,000 transactions per second, or 510,000 messages per second.

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



3 Channel Compression use of on-chip compression accelerator

Introduced with z15, on-chip compression replaces the zEnterprise Data Compression (zEDC) feature on the IBM z14 and earlier platforms.

The z15 on-chip compression module implements DEFLATE, gzip and lzip algorithms and works in the following modes:

1. Synchronous execution for problem state
2. Asynchronous optimization for large operations under z/OS.

The blog article "Taking a peek under the Hood of Compression on z15" discusses the changes to compression on z15 in some detail.

MQ supports the use of compression, whether using hardware via on-chip (z15), zEDC (z14 and earlier) or software, for channel compression with the setting of the channel attribute `COMPMSG(ZLIBFAST)`.

MQ channel compression is performed using the synchronous execution, which reduces the latency of the switch to/from the PCIE-based zEDC processor on z14.

Additionally the threshold for performing compression in hardware on z15 has been lowered to 1KB and is no longer configurable via the `PARMLIB(IQPPRMxx)` member.

MQ will fall back to software for channel compression with `COMPMSG(ZLIBFAST)` when:- the message falls below the inflate / deflate thresholds, or when hardware (zEDC on z14 or earlier) is not available.

These improvements can make a significant difference in the performance of workload flowing over MQ channels.

With highly compressible messages we have seen a decrease in transaction cost of up to 42% when compared with the equivalent workload on z14, with an increase to the transaction rate of up to 90%.

When combined with TLS-protected channels and a frequent renegotiation of the secret key, we have seen up to a 60% improvement in throughput over a low-latency network.

Channel compression using ZLIBFAST

In this example, the workload runs a request/reply model using 32KB non-persistent messages between 2 z/OS queue managers.

The queue managers are located on separate LPARs on the same physical machine, linked by a dedicated low-latency 10Gb network.

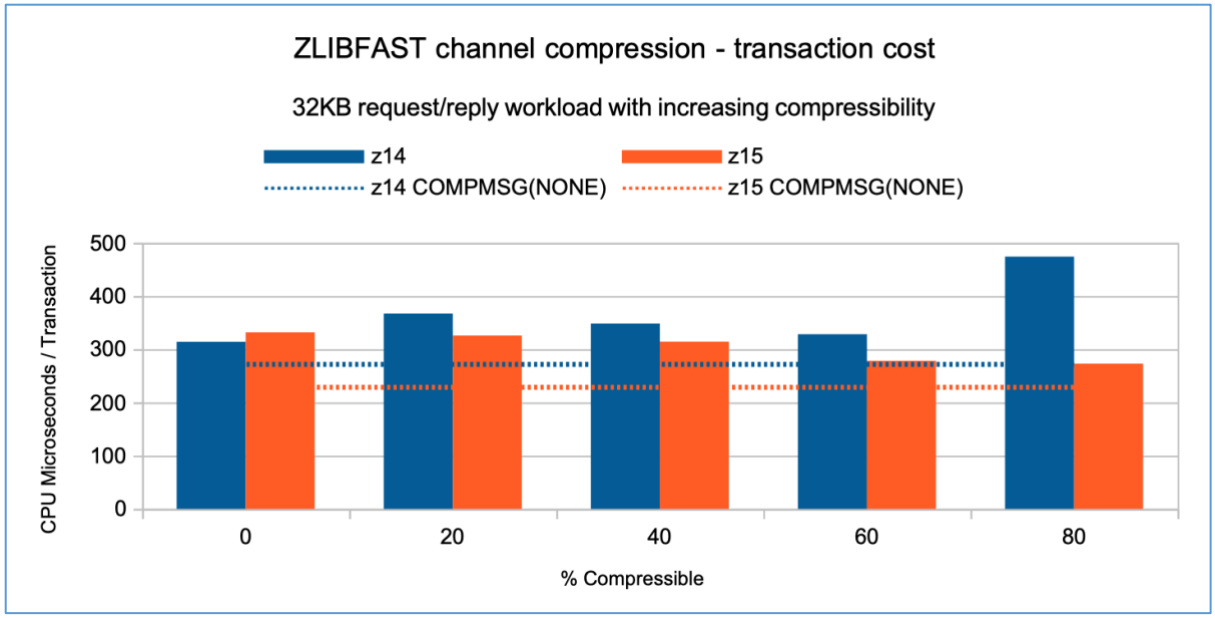
In this request/reply model, the message is compressed and inflated twice – once for the request message and again for the reply message.

In each case, the channels are configured with:

- `COMPMSG(ZLIBFAST)`

The following chart compares the transaction cost when the message payload increases in compressibility for both z14 and z15.

The chart additionally shows the equivalent transaction cost when `COMPMSG(NONE)` is specified, i.e. no compression is attempted.

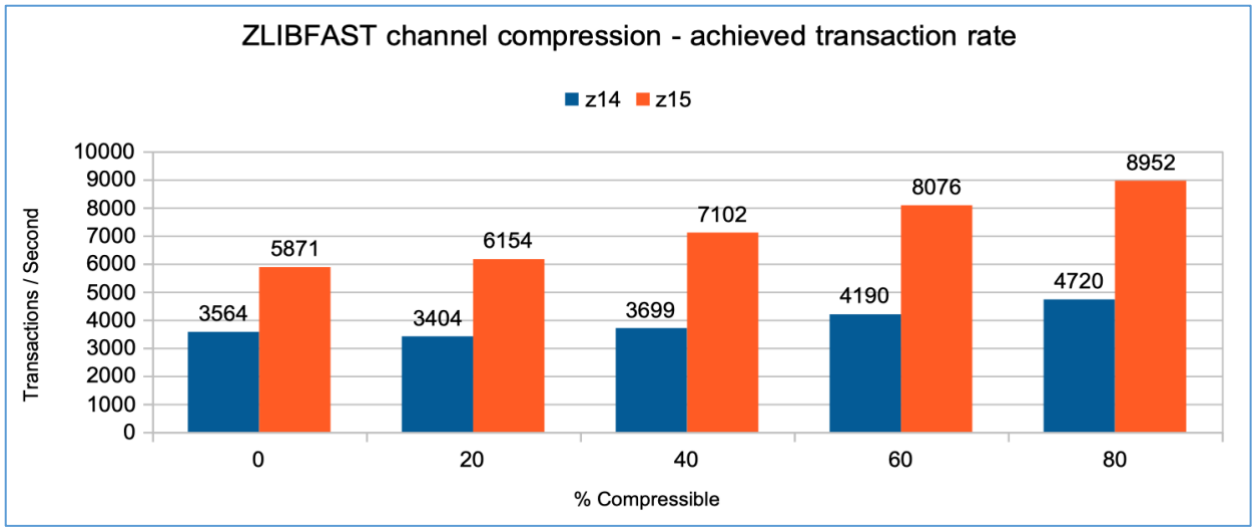


Notes on transaction cost:

1. z15 costs are up to 43% lower than the equivalent z14 measurement, with more compressible messages demonstrating the largest difference.
2. z15 costs for “incompressible” messages are higher than the equivalent z14 measurement. This is due to the z14 measurement being unable to compress the payload, whereas on z15, the message payload was 3% compressible and therefore needed to be inflated by the receiving channel initiator.
3. The significant increase in z14 cost between 60 and 80% compressible is due to the compressed message being too small to inflate in hardware, instead having to be inflated in software. The lower thresholds on z15 means that all of the payload is compressed and inflated in hardware.

4. The measurements with no compression always demonstrated lower transaction cost than compressing the message payload, however compressing highly compressible messages on z15 showed parity with uncompressed messages on z14.

In the following chart, the achieved transaction rates are compared between z14 and z15.



Notes on transaction rate chart:

1. The z15 results throughput is 65 to 90% higher than the z14 equivalent, demonstrating the reduction in latency from on-chip compression.

Channel compression using ZLIBFAST on TLS channels

In this example, the workload runs a request/reply model using 32KB non-persistent messages between 2 z/OS queue managers.

The queue managers are located on separate LPARs on the same physical machine, linked by a dedicated low-latency 10Gb network.

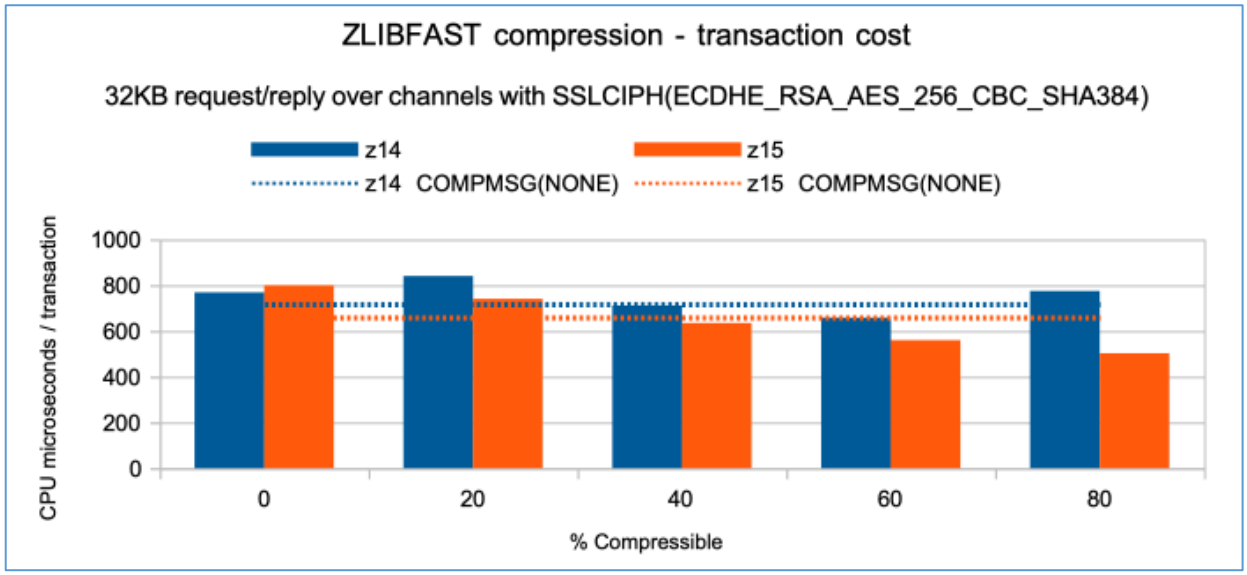
In this request/reply model, the message is compressed and inflated twice – once for the request message and again for the reply message.

In each case, the channels are configured with:

- `SSLCIPH(ECDHE_RSA_AES_256_CBC_SHA384)`
- `SSLRKEYC(1MB)`
- `COMPMSG(ZLIBFAST)`

The following chart compares the transaction cost when the message payload increases in compressibility for both z14 and z15.

The chart additionally shows the equivalent transaction cost when `COMPMSG(NONE)` is specified, i.e. no compression is attempted.

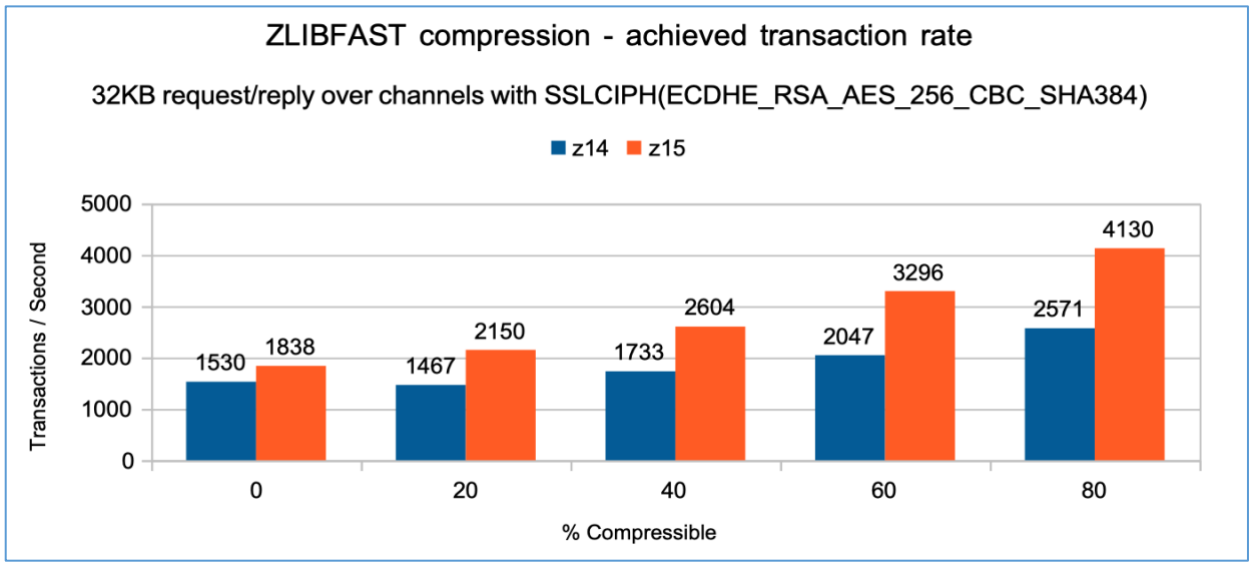


Notes on transaction cost chart:

1. z15 costs are up to 35% lower than the equivalent z14 measurement, with more compressible messages demonstrating the largest difference.
2. z15 costs for “incompressible” messages are higher than the equivalent z14 measurement. This is due to the z14 measurement being unable to compress the payload, whereas the z15 measurement was able to compress by 3% and as a result needed to inflate the message.
3. Both z14 and z15 demonstrates cost savings at 40% compressible over the COMPMSG(NONE) measurement.

4. Message compression takes place before secret key negotiation, so with compressible messages, more messages can flow over the channels before hitting the SSLRKEYC threshold.

In the following chart, the achieved transaction rates are compared between z14 and z15.



Notes on transaction rate chart:

2. The z15 results throughput is 20 to 60% higher than the z14 equivalent, demonstrating the reduction in latency from on-chip compression.

4 Archive log compression using on-chip compression accelerator

As mentioned in the previous section, on-chip compression replaces the zEnterprise Data Compression feature on the IBM z14 and earlier platforms.

Archive log compression uses the ***asynchronous*** mode of compression.

Asynchronous compression and decompression activity can be reported using RMF's Extended Asynchronous Data Mover (EADM) report.

RMF Monitor I collects utilization information of IOP resources used by EADM-compression. The returned utilization information is stored in SMF 78 subtype 3 records.

In addition the RMF postprocessor and Monitor III SCM activation reports are renamed EADM Activity reports and, provided that SMF 74 subtype 10 records have been collected, can be formatted by the ERBRMFPP program using the "REPORTS(EADM)" option.

Previously we have written about using zEDC with MQ archive logs in terms of potentially reducing storage occupancy in both the blog "Reducing storage occupancy with IBM zEnterprise Data Compression (zEDC) on IBM MQ for z/OS" and the "IBM MQ for z/OS on z14 performance" report, but this section provides an update discussing the benefits of z15.

For these measurements we have re-used the process and configurations detailed in the earlier performance report 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 log compression on MQ archive logs on the z15 performance system was to reduce the load on the disk subsystems' cache such that the MQ log performance was improved by up to 82% in terms of log write rate over the uncompressed log write rate. This improvement did come at a cost to CPU and an increase in recovery time, and the impact will depend on the compressibility of the data.

Impact of compressed archives on regular workload performance

For the range of message compressibility used (0-80% compressible), we measured increased costs in queue manager TCB relating to the archive process as detailed in the following table as a result of compressing the archive logs:

Message Size	4KB	32KB	1MB	4MB
Increase in QM TCB over uncompressed measurement	0%	Up to 5%	8-13%	8-14%
Increase in peak log throughput	1%	Up to 3%	32-65%	45-82%

In the worst case, the cost of archiving increase by 14%, which was for an incompressible message payload of 4MB messages.

The side effect of highly compressible messages resulting in lower load on the I/O subsystem meant that the impact to the peak messaging rate increased from 282MB/sec to 520MB/sec per log, when achieving 80% compression on 4MB messages.

Recovery from compressed archive logs

The rate at which MQ is able to recover from compressed archive logs is significantly less that the rate at which MQ can recover from uncompressed archives.

	Uncompressed Archives	**Compressed Archives**
Recovery rate (MB/sec)	106	40
Cost per MB (CPU microsecond)	930	1451

- Recovery rate of the compressed archives is 38% of the uncompressed archives.
- Recovery cost per MB of archive logs is 1.6 times that of uncompressed archives.

You should consider whether the benefits of reduced storage occupancy and less load on your I/O subsystems for regular workload outweighs any impact to recovery schedules.

5 Improvements from Crypto Express 7S

As mentioned in the previous section, the z15 benefits from some significant improvements in the cryptographic area – primarily in Crypto Express 7S (CEX7S).

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 data set encryption is reported separately in section "Data set Encryption".

The comparisons are between:

- IBM z14 with Crypto Express6S (CEX6S)
- IBM z15 with Crypto Express7S (CEX7S)

Channels protected using SSLCIPH specifications

When performance testing channels protected with cipher specifications, we limit our testing to the following 4 ciphers:

1. `TLS_RSA_WITH_AES_256_CBC_SHA256`
2. `TLS_RSA_WITH_AES_128_GCM_SHA256`
3. `ECDHE_RSA_AES_256_CBC_SHA384`
4. `ECDHE_ECDSA_AES_256_CBC_SHA384`

For these ciphers, we run in 2 modes: frequent secret key negotiation and no secret key re-negotiation.

The cost of the secret key negotiation is largely in the MQ channel initiator address space, but some of the cost can be offloaded to the Crypto Express hardware.

Crypto Express 7S "CEX7S" offers reduced execution time compared to earlier generations of Crypto Express.

According to the RMF Cryptographic report, the execution times have decreased by up to 13% for both coprocessor and accelerator type features.

In our measurements, the TLS prefixed ciphers were able to exploit both coprocessor and accelerator, unlike the ECDHE prefixed ciphers that were only able to use coprocessor.

In terms of MQ channel address space cost, the `TLS_RSA` prefixed ciphers are considerably lower cost at the time of secret key negotiation.

Cost of secret key negotiation (CPU milliseconds) in MQ channel initiator address space:

Cipher	z14	z15
`TLS_RSA_WITH_AES_256_CBC_SHA256`	0.66	0.64
`TLS_RSA_WITH_AES_128_GCM_SHA256`	0.65	0.7
`ECDHE_RSA_AES_256_CBC_SHA384`	4	3.8
`ECDHE_ECDSA_AES_256_CBC_SHA384`	3.9	3.8

When running with no secret key re-negotiation outside of channel start, the 4 named ciphers deliver comparable performance at similar cost on z15.

For the purposes of this section, we compare performance results using all of these named cipher specifications using 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.

For simplicity and clarity we are only showing the performance with a single pair of inbound and outbound channels.

As mentioned previously, the measurements are run in 2 configurations:

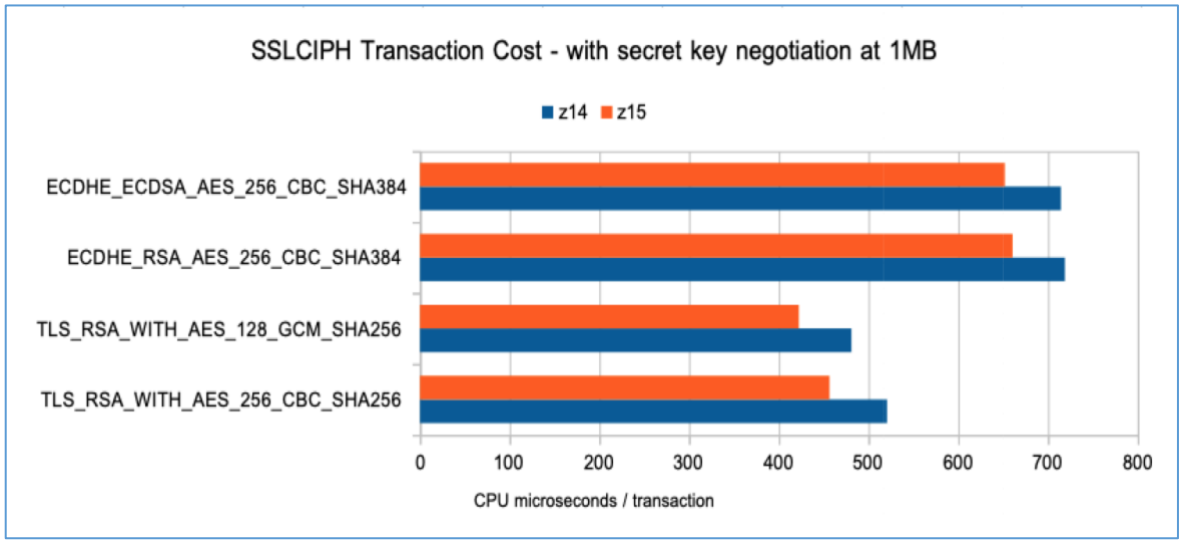
1. Frequent negotiation of secret key:
    a. Negotiate the secret key every 1MB that passes over the channel, by setting SSLRKEYC(1048576).

   b. Demonstrates the benefits of CEX7S and CPACF.
2. No renegotiation of secret key:
   a. Negotiate the secret key at channel start only – with the channels remaining active for the duration of the workload.
   b. Demonstrates the benefits of CPACF only.

Frequent Re-negotiation of secret key

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.

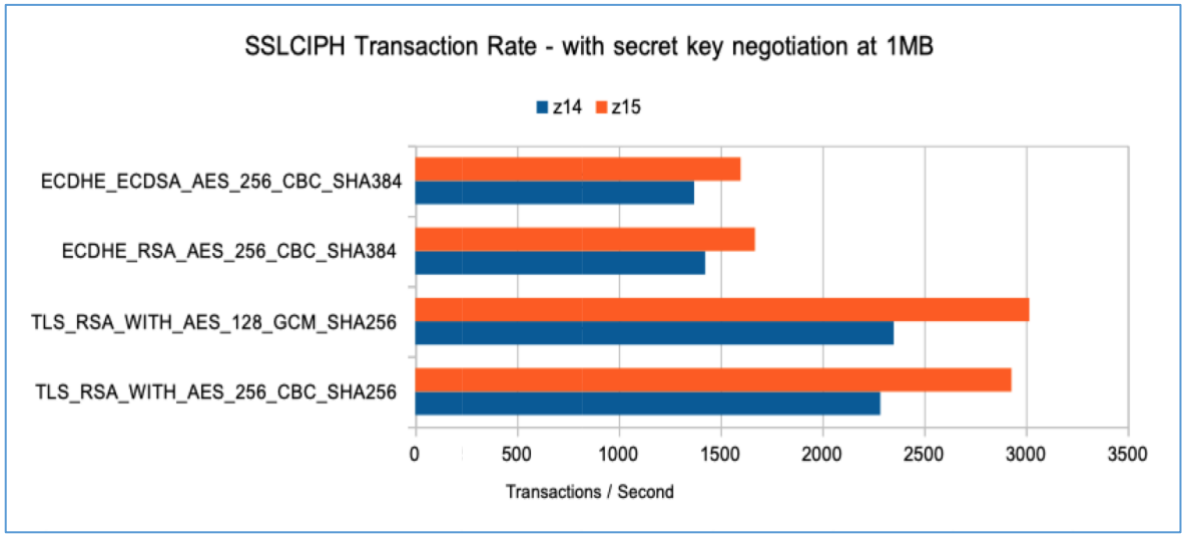
The following chart compares the cost of the workload between z15 and z14.



The transaction cost chart shows that in our measurements, that z15 reduced the cost of the workload as below:

`TLS_RSA` prefixed ciphers: 12% lower on z15.
`ECDHE` prefixed ciphers: 8% lower on z15.

In the second chart, the transaction rate achieved by the workloads is shown.



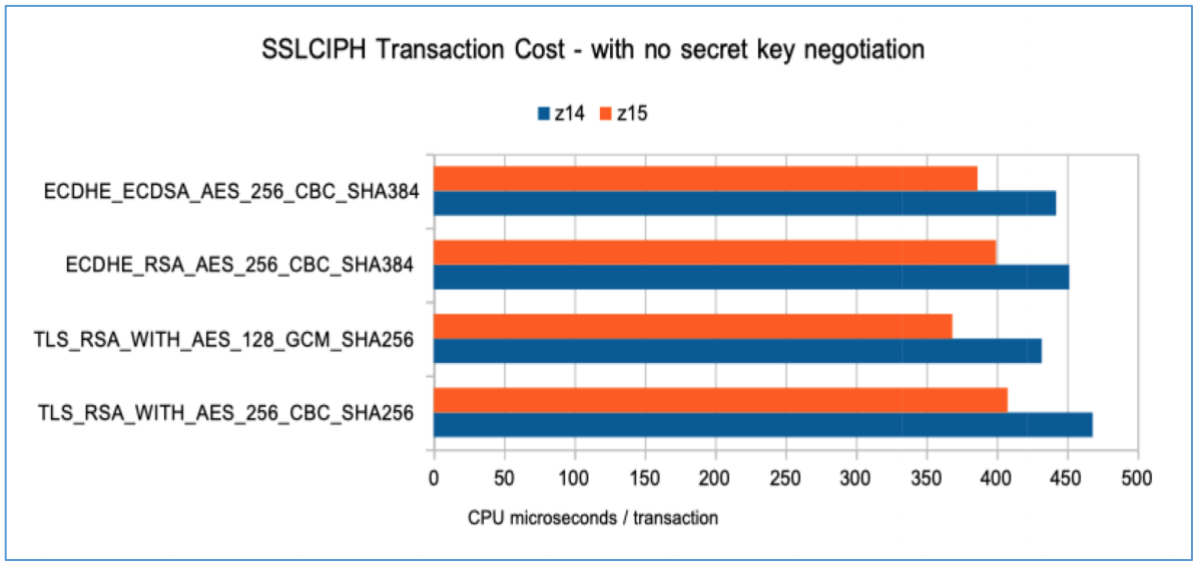
Across the workloads, we saw improvements in transaction rates over the equivalent z14 measurements, as detailed below:

`TLS_RSA` prefixed ciphers:	28% higher on z15.
`ECDHE` prefixed ciphers:	17% higher on z15.

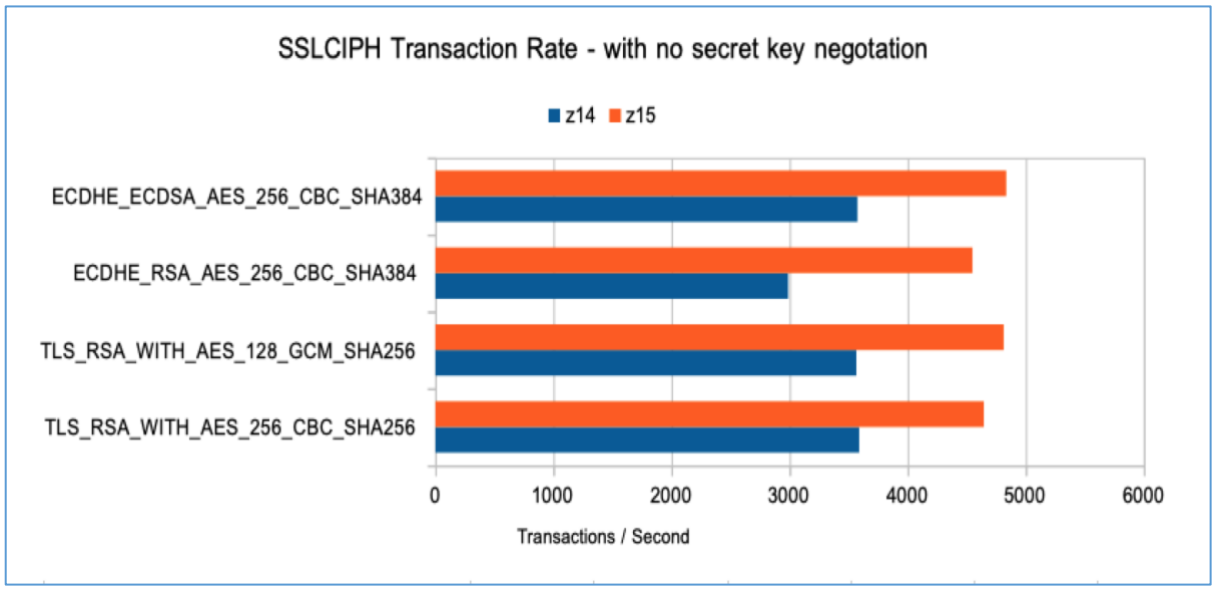
No renegotiation of secret key

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 z14 and z15. 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 z15 is considerably lower cost for these types of workloads than z14 – our tests show a reduction in transaction cost of up to 15%.



The second chart shows the transaction rate achieved when the secret key is negotiated at channel start. The transaction rate is up to 53% higher on z15.



TLS/SSL channel start costs

The rate and CPU cost at which channels can be started varies with the number of channels represented in the SYSTEM.CHANNEL.SYNCQ.

A channel is represented in the SYSTEM.CHANNEL.SYNCQ if it has ever been started. It will remain represented until its definition is deleted. For this reason we recommend that redundant channel definitions be deleted.

Whilst many users do not start channels with any great frequency, there may still be significant sender channel restart activity after a channel initiator failure.

Whenever an TLS-enabled channel pair[1] is started, a cryptographic handshake is performed which establishes the authenticity of the channel partners and dynamically generates a secret cryptographic encryption key. This cryptographic handshake increases both the CPU consumption and the elapsed time for the channel start.

On our 8561 system configured with 3 dedicated processors, we have found the additional TLS costs to be somewhat dependent of the cipher specification used. With 4000 channel pairs in SYSTEM.CHANNEL.SYNCQ:

Cipher	Channels started per second	CPU cost milliseconds / channel
TLS_RSA_WITH_AES_256_CBC_SHA256	174	2.21
ECDHE_RSA_AES_256_CBC_SHA384	103	3.9
ECDHE_ECDSA_AES_256_CBC_SHA384	98	3.76

For the TLS_RSA_WITH_AES_256_CBC_SHA256 cipher, this represents an 18% improvement over the equivalent z14 measurement.

For the ECDHE prefixed ciphers, this represents between 9 to 16% improvement over the equivalent z14 measurement.

Note that channel start costs of TLS-enabled channels are significantly higher than the costs incurred at time of secret key negotiation.

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:

[1] In this example, a channel pair is one CHLTYPE(SDR) and one CHLTYPE(RCVR).

- 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.

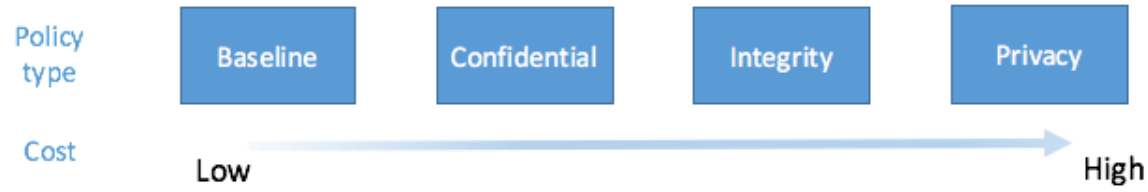
Significant performance improvements to AMS protection were applied to IBM MQ on z/OS in Continuous Delivery Release version 9.0.1 as documented in the performance report.

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.

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



The performance of our tests with queues protected by AMS policies has not significantly changed when moving from z14 to z15. Small message workloads have seen the largest improvement in terms of both cost reduction and throughput increase.

The following tables indicate the change in performance between z14 and z15.

% change in transaction cost from z14 to z15:

Message Size	Integrity	Privacy	Confidential
Small	-8	-7.8	-12
Medium	-6.6	-6.9	-8.8
Large	-4.5	-6.3	-7.4

% change in throughput from z14 to z15:

Message Size	Integrity	Privacy	Confidential
Small	+8.5	+9.4	+13
Medium	+7.4	+8.4	+9.8
Large	+4.3	+7.25	+4.5

Appendix A – Test Environment

Measurements were performed using:

The IBM MQ performance sysplex ran measurements on:

- **IBM z14 (3906-7H1) – 4 CPC drawers**
- **IBM z15 (8561-7J0) – 4 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 v2r3.
- Db2 for z/OS version 12 configured for MQ using Universal Table spaces.
- MQ queue managers:
  - configured at MQ V9.1.4.
  - 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