z/OS Communications Server TLS Performance Update

Created by: Christopher Nyamful
cnyamfu@us.ibm.com

Trademarks, Notices, and Disclaimers

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries:

BigInsights	HyperSwap	System z10*
BlueMix	IBM*	Tivoli*
CICS*	IBM (logo)*	UrbanCode
COGNOS*	IMS	WebSphere*
Db2*	Language Environment*	z13
DFSMSdfp	MQSeries*	z14
DFSMSdss	Parallel Sysplex*	z15
DFSMShsm	PartnerWorld*	z16
DFSORT	RACF*	zEnterprise*
DS6000*	Rational*	z/OS*
DS8000*	Redbooks*	zSecure
FICON*	REXX	z Systems
GDPS*	SmartCloud*	z/VM*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies:
Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation
or its subsidiaries in the United States and other countries.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce and is registered in the U.S. Patent and Trademark Office.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
Red Hat is a trademark of Red Hat Inc, an IBM Company.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.
* Other product and service names might be trademarks of IBM or other companies.

Notes:
Performance results are based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon 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 throughput improvements equivalent to the performance ratios stated here. IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions. This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area. All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only. Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, 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. Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography. This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g, zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the “Authorized Use Table for IBM Machines” provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html (“AUT”). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Contents

Preface

The performance information in this publication was measured on an IBM z16 in a controlled environment. Many factors can impact the outcome of performance results. Therefore, no assurance can be given that an individual user will achieve the same results shown in this document. The results described herein are presented for informational purposes. Actual performance and security characteristics will vary depending on individual configurations and conditions.

The Central Processor Unit (CPU) numbers listed includes only z/OS host networking related CPU overhead (including dispatching cost) on general purpose CPU from the network device driver layer up through the application socket layer. The socket application used to drive the benchmarks for this publication has no application logic. With typical production workloads, network related cost is a small fraction of the overall application transaction cost (approximately 5%).

Terminology

Please reference these terms here [2] for detailed explanation.

AES – Advanced Encryption Standard

CPACF – CP Assist for Cryptographic Function

DHE – ephemeral Diffie-Hellman

ECDHE – ephemeral Elliptic Curve Diffie-Hellman

ECC – Elliptic Curve Cryptography

ECDSA – Elliptic Curve Digital Signature Algorithm

GCM – Galois Counter Mode

HDKF – HMAC-based Extract-and-Expand Key Derivation Function

ICSF – Integrated Cryptographic Service Facility

RSA – Rivest-Shamir-Adleman

SHA – Secure Hash Algorithm

SSL – Secure Sockets Layer

TLS – Transport Layer Security

Background

Application Transparent Transport Layer Security (AT-TLS) is a part of z/OS Communications Server that provides TLS protection to TCP/IP applications running on z/OS. AT-TLS is a policy-based technology and can be fully transparent to applications needing TLS protection. System SSL is a z/OS native TLS implementation that provides TLS protection to network traffic. AT-TLS is built to make direct calls to System SSL on behalf of Language Environment (LE) applications running on z/OS. The integration between System SSL and AT-TLS is optimized to ensure it performs as well or better than a direct integration from an application program to System SSL.

This publication shows the network performance impact of selected TLSv1.2 [4] and TLSv1.3 [5] protocol cipher suites. We highlight improvements made in ICSF and System SSL. The performance numbers shown in this paper are for networking only and do not include the business logic found in typical applications.

TLS Handshake

TLS processing can be CPU-intensive based on the nature of algorithms in the cipher suite. Part of this processing is in the TLS handshake, which takes place at the beginning of each TLS session. The handshake phase involves asymmetric operations (digital signature operations, Diffie-Hellman key exchange operations, etc), which can impact network performance (latency, CPU). Some of these operations benefit from hardware acceleration (CPACF and Crypto Express adapters).

Benchmark Environment

Benchmark environments consist of two dedicated logical partitions (LPARs), with dedicated CPUs and Crypto Express cards, as shown in Figure 1. The measurements are done in a controlled environment, hence there is no guarantee your network cost will be the same as shown in this report.

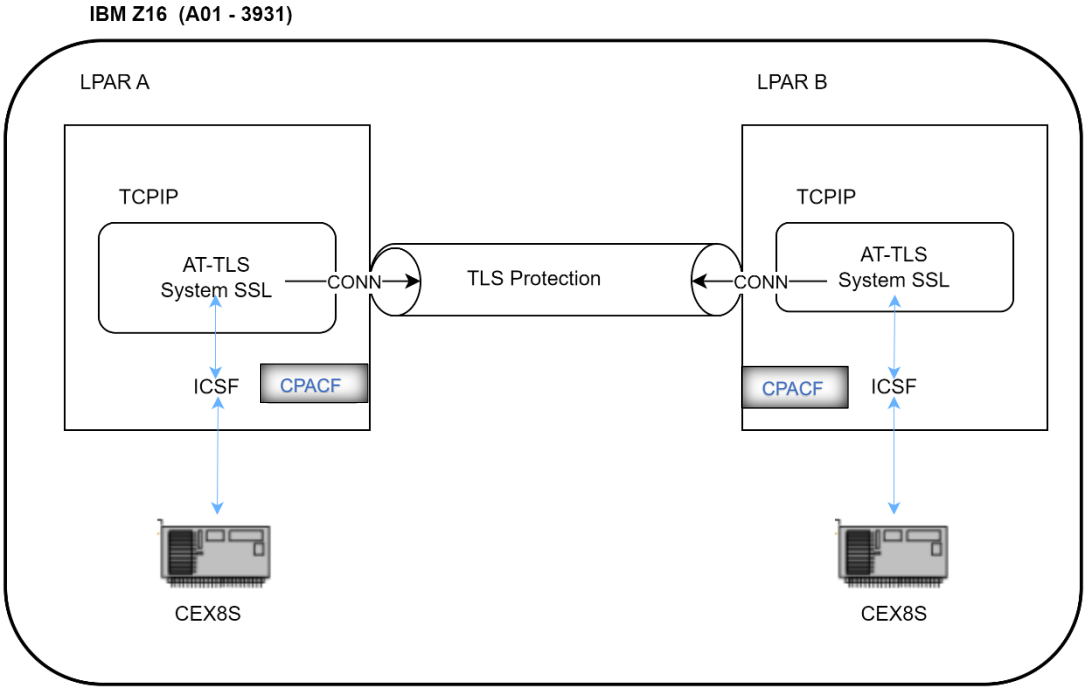


Figure 1. Controlled environment with dedicated LPARs, CPs, Crypto Express cards, etc.

Hardware:

CPC :

z16 (A01 – 003931)

#CPU :

2 LPARs with 4 dedicated general-purpose processors (GCPs) (8 total)

Network Interface :

OSA Express 7S 10Gbit

Crypto Adapter :

1 dedicated Crypto Express 8S (level=8.0.71z) per LPAR
Coprocessor mode – for secure key encrypted operations
Accelerator mode – for clear key acceleration [1]

Software:

z/OS Release: 3.1

ICSF Level: HCR77E0 (APAR OA64635)

System SSL: Service (APAR OA63252)

Cipher Suites:

We used a blend of selected TLSv1.2 and TLSv1.3 protocol cipher suites to highlight the performance of Elliptic Curve Cryptography (ECC)-based key exchange, and different digital signature algorithm pairs. All the ciphers are AES-256-GCM based and use SHA-384 for stronger hashing and message integrity. Below is the list and description of ciphers used in our performance measurement:

TLSv1.2 cipher suites:

C030 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

- *256-bit AES in Galois Counter Mode encryption with 128-bit AEAD message authentication and ephemeral ECDH key exchange signed using RSA certificate*

C02C TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384[1]

- *256-bit AES in Galois Counter Mode encryption with 128-bit AEAD message authentication and ephemeral ECDH key exchange signed using ECDSA certificate*

009D TLS_ RSA_WITH_AES_256_GCM_SHA384

- *256-bit AES in Galois Counter Mode encryption with 128-bit AEAD message authentication and RSA key exchange*

009F TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

- *256-bit AES in Galois Counter Mode encryption with 128-bit AEAD message authentication and ephemeral Diffie-Hellman key exchange signed using RSA certificate* [2]

TLSv1.3 cipher suites:

1302 TLS_AES_256_GCM_SHA384

- *256-bit AES in Galois Counter Mode encryption with 128-bit AEAD authentication and HKDF (HMAC-based Extract-and-Expand Key Derivation Function) with SHA384* [2]

Note: *The SHA384 in all these cipher suites indicates the TLS PseudoRandom Function (PRF) will be based on a SHA-384 bit hash*

Digital Certificates:

Most of the measurements were done using an RSA self-signed server certificate (signed with its own private key) with a key size of 2048 bits. Other measurements were done using a self-signed ECDSA server certificate with an ECC key size of 256 bits. TLS client authentication is not used in these measurements. A snippet of the certificate information is shown below:

Signing Algorithm: sha256RSA
Key Type: RSA
Key Size: 2048

Signing Algorithm: sha256ECDSA
Key Type: NIST ECC
Key Size: 256

Client ECurve: secp256r1 (0023)

[1] Measurements with this cipher uses a key type of NIST ECC and key size 256-bit

Workload:

To highlight the performance impact of the TLS handshake process, we use a network connection benchmark, that simulates short-lived connections to highlight the performance of the TLS handshake process. In our measurements, we open 40 concurrent connections between a TLS client and TLS server. Each concurrent connection client:

- Opens a TCP connection
- Sends 64 bytes and receives 8000 bytes
- Closes the TCP connection
- Repeats this process in a loop for the test period in order to generate a large number of TLS handshakes

Notes:

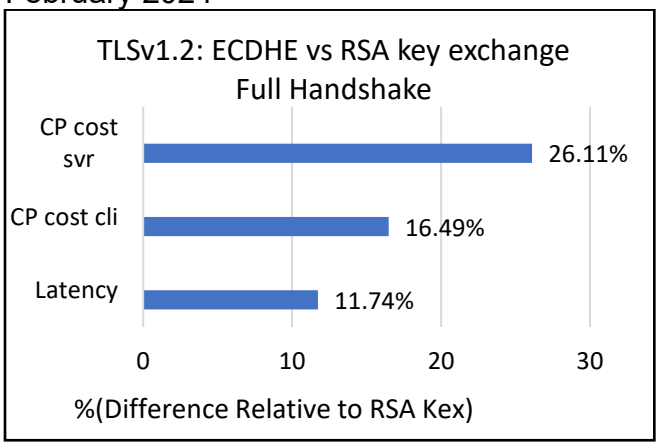
- The CPU measurements gathered for this lightweight-application benchmark includes all z/OS host networking related CPU overhead up through the application socket layer
- The socket applications used have no application logic
    - The networking related CPU cost equals the entire benchmark CPU cost
    - In typical workloads, networking related CPU cost is a fraction of overall application transaction cost (the benchmark shows the worst-case scenario from a networking related CPU perspective)

ECC-Based Cipher Performance

ECDHE key exchange vs. RSA key exchange

The C030 cipher supports ephemeral elliptic curve Diffie-Hellman key exchange, which allows for perfect forward secrecy (PFS), hence more secure compared to the RSA key exchange ciphers. ECDHE key exchange requires several elliptic curve Diffie-Hellman operations, including the generation of a brand-new key pair. The addition of these operations to achieve PFS add additional CPU costs - up to a 26% increase for full handshake processing in this comparison. This increased CPU cost with PFS can be significantly reduced with the enablement of TLS session resumption (abbreviated handshakes) as shown in Figure 2. See *Sysplex-wide Session Resumption* below for more details.

(C030) TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
vs. (009D) TLS_RSA_WITH_AES_256_GCM_SHA384



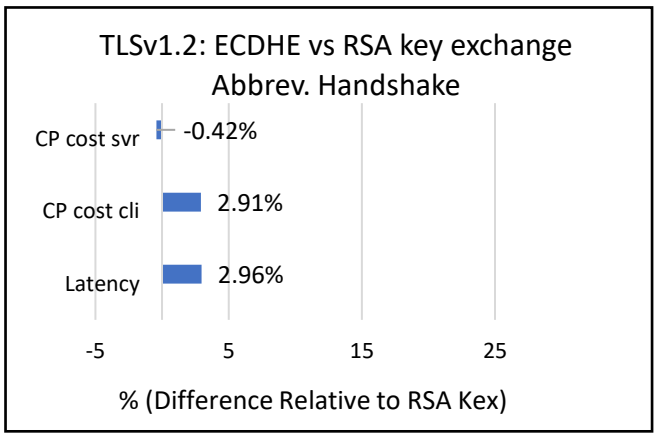


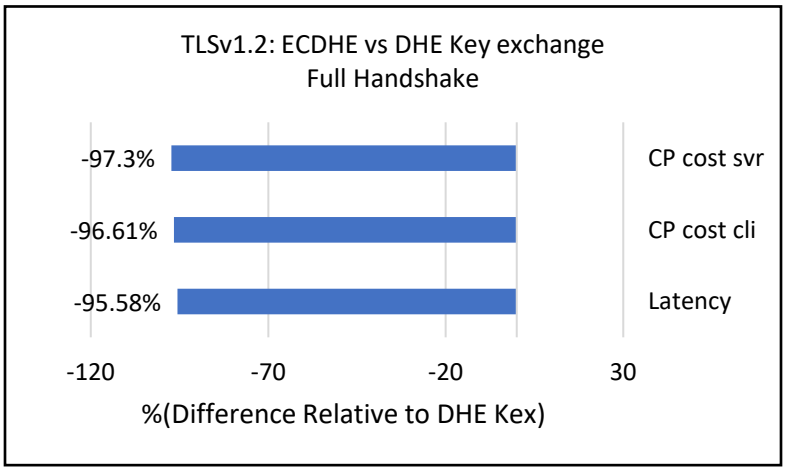
Figure 2. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 vs TLS_RSA_WITH_AES_256_GCM_SHA384 performance

Conclusion: *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 provides you with a much stronger key exchange mechanism and has similar performance impact as RSA key exchange when using TLS session resumption.*

ECDHE key exchange vs. DHE key exchange

The performance results in Figure 3 show how an ECDHE-based cipher performs against the Finite Field DHE-based cipher. Note that on z/OS, all DHE key exchange processing is performed in software (no hardware acceleration) and therefore has significantly higher CPU consumption compared to ECDHE key exchange, which is performed on CPACF. Most of the processing done in software for the DHE key exchange is removed when you enable TLS session resumption.

(C030) TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
vs. (009F) TLS_DHE_RSA_WITH_AES_256_GCM_SHA384



- Up to 97% reduction in CPU cost per transaction compared to DHE key exchange
- Significant reduction (95%) in transactional latency compared to DHE key exchange

Figure 3. Elliptic Curve DHE vs Finite Field DHE based cipher performance

Conclusion: ECDHE key exchange is vastly faster and less costly than DHE. If at all possible, favor ECDHE cipher suites over those that use DHE.

RSA vs. ECDSA Certificate Performance

The result in Figure 4 compares the performance of using a certificate with an RSA key size of 2048 bits to a certificate with an ECDSA key size of 256. The ECC-key size 256-bit is much smaller but provides equivalent encryption security compared to the RSA key size of 2048-bit. RSA operations are being offloaded to the Crypto Express adapter, and the ECDSA operations are performed by the CPACF. ICSF's performance is relatively equivalent when using available hardware for RSA and ECDSA digital signature generation and verification [3], hence the networking performance cost of these operations is relatively equivalent.

(C030) TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
vs. (C02C) TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

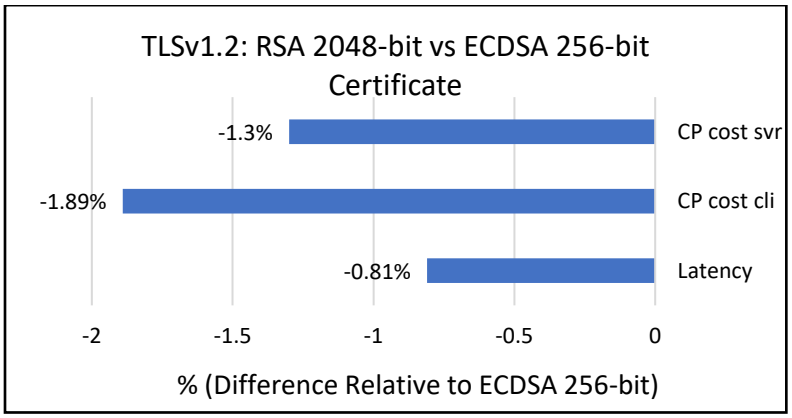


Figure 4. RSA 2048-bit vs ECDSA 256-bit certificate performance

Conclusion: The performance difference doesn't justify converting your RSA keys to ECC in the near term.

TLSv1.3 vs. TLSv1.2

TLSv1.3 has more required key derivation operations compared to TLSv1.2, and it uses the HKDF key derivation algorithm which is primarily implemented in software [3]. These operations enhance the security of TLSv1.3 but also make it more compute-intensive compared to the TLSv1.2 protocol. The Language Environment (LE) variable option, **HEAPPOOLS64**, is enabled on the TCP/IP stack in both measurements to remove any LE heap latch contention during TLS secure handshake. The HEAPPOOLS64 option becomes a factor with TLSv1.3 under very low handshake volumes, unlike TLSv1.2, so it is imperative that this option be enabled when using TLSv1.3. When HEAPPOOLS64 is enabled, TLSv1.3 performance is very competitive to TLSv1.2, with just 16% increase in network CPU cost, as shown Figure 5. z/OS Communications Server **APAR PH59425** ensures that the LE HEAPPOOLS64 option is always enabled for AT-TLS.

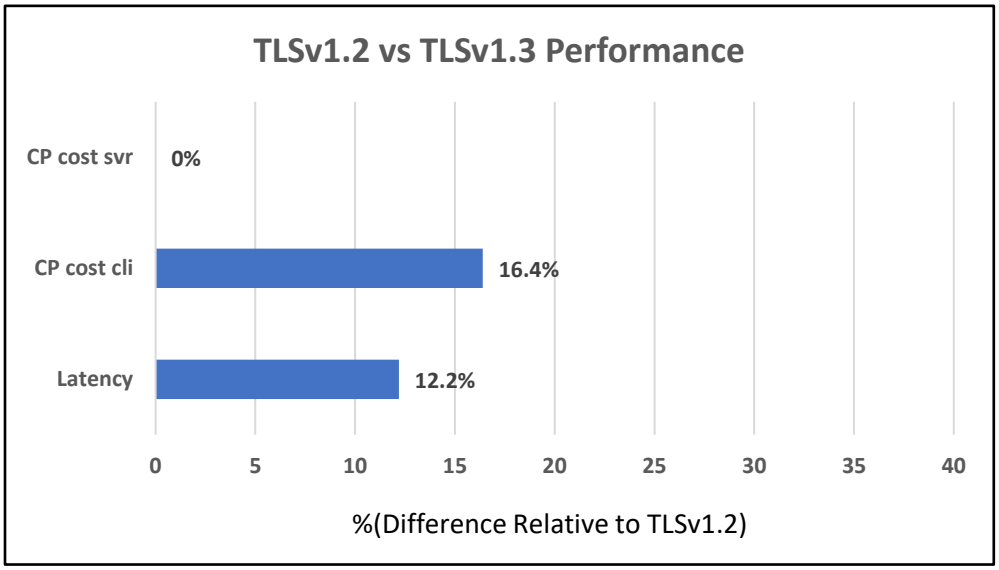
Note: *In this measurement, we tried to impose as much similarity as possible regarding the algorithms being used between TLSv1.2 and TLSv1.3 in order to isolate the inherent differences between the protocol performance. Users who are migrating from TLSv1.2 protocol to TLSv1.3 protocol might have been using TLSv1.2 cipher suites that do not require PFS (for example, TLS_RSA_WITH_xxxx). In those cases, the difference in CPU consumption and transactional latency, will be even more pronounced than illustrated here.*

TLSv1.2

```
ConnID: 00000037
  JobName:      APF1
  LocalSocket:  10.67.170.128..5001
  RemoteSocket: 10.67.170.126..1034
  SecLevel:     TLS Version 1.2
  Cipher:       C030 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
  KeyShare:     N/A
  CertUserID:   N/A
  MapType:      Primary
  FIPS140:      Off
  SessionID:    00000094 0943AA7E 040A0000 00000000
                00000000 00000000 65AAA842 00000009
   SignaturePairs:             0804 TLS_SIGALG_SHA256_WITH_RSASSA_PSS
                               0401 TLS_SIGALG_SHA256_WITH_RSA
```

TLSv1.3

```
ConnID: 000000C6
  JobName:      APF7
  LocalSocket:  10.67.170.128..5001
  RemoteSocket: 10.67.170.126..1065
  SecLevel:     TLS Version 1.3
  Cipher:       1302 TLS_AES_256_GCM_SHA384
  KeyShare:     0023 secp256r1
  CertUserID:   N/A
  MapType:      Primary
  FIPS140:      Off
  SessionID:    00000035 0943AA7E 04290000 00000000
                00000000 00000000 65AA9742 00000009
   ClientKeyShareGroups:       0023 secp256r1
   ServerKeyShareGroups:       0023 secp256r1
   SignaturePairs:             0804 TLS_SIGALG_SHA256_WITH_RSASSA_PSS
```



- LE runtime option **HEAPPOOLS64** enabled on the stack for TLSv1.2 and TLSv1.3

- All the increased CPU cost of the TLSv1.3 handshake is seen on the client side – there is no net increase in the server-side CPU

Figure 5. When using similar cryptographic algorithms, TLSv1.3 performance is competitive with that of TLSv1.2

TLS Session Resumption Performance (abbreviated handshake)

Session ID or session ticket caching reduces the number of full handshakes that is required in the TLS environment. TLSv1.2 session ID caching and TLSv1.3 session tickets reduce the number of full TLS handshakes in cases where clients repeatedly establish TLS connections to the same server. By resuming a TLS session with a valid TLSv1.2 session ID or a TLSv1.3 session ticket, the TLS server avoids many of the costly asymmetric operations required in a full handshake. The use of these "abbreviated handshakes" significantly reduces CPU cost and latency. Figure 6 shows the benefits of enabling TLS session resumption for selected TLSv1.2 protocol ciphers (up to 50% savings in CPU cost), and Figure 7 shows up to 30% CPU savings for TLSv1.3 protocol.

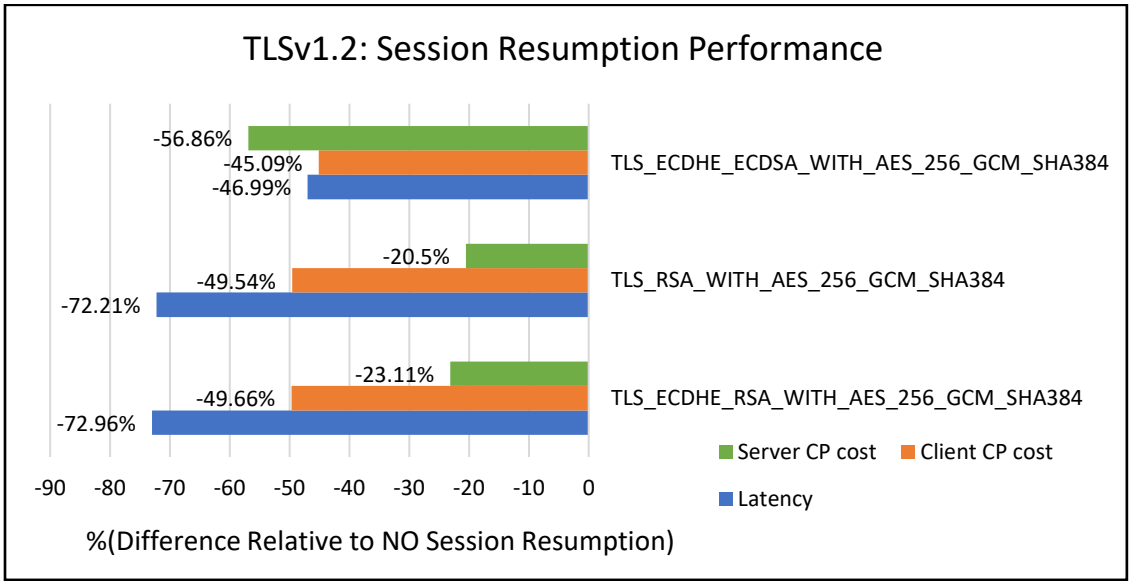


Figure 6. Performance benefits (CPU cost & Latency) of enabling TLS session resumption for selected TLSv1.2 ciphers

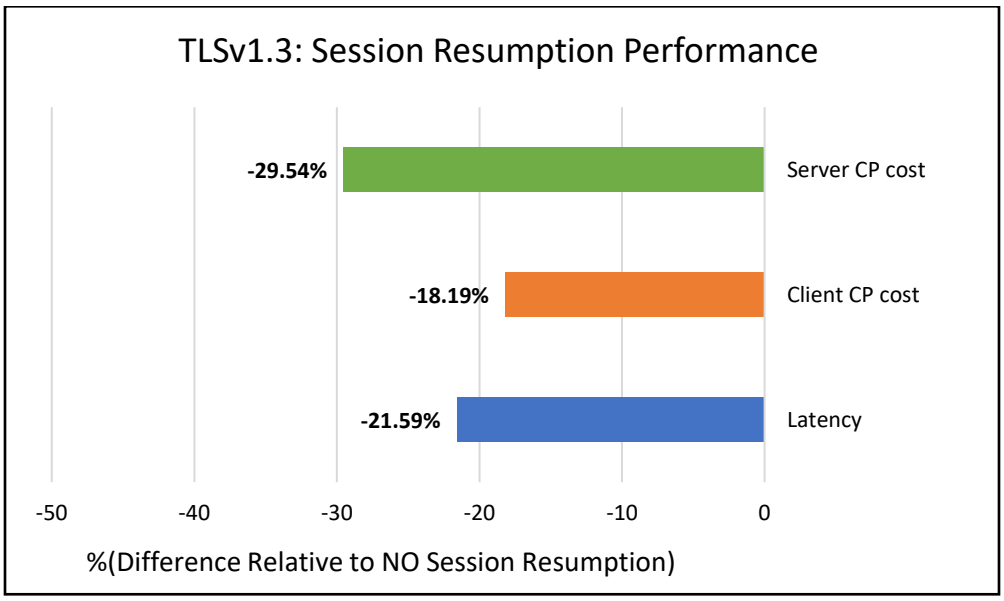
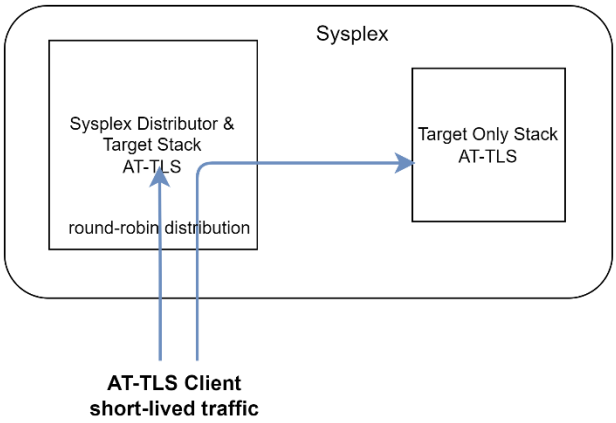


Figure 7. Performance benefits (CPU cost & Latency) of enabling TLS session resumption for TLSv1.3 (TLS_AES_256_GCM_SHA384 cipher with RSA key type, key size 2048)

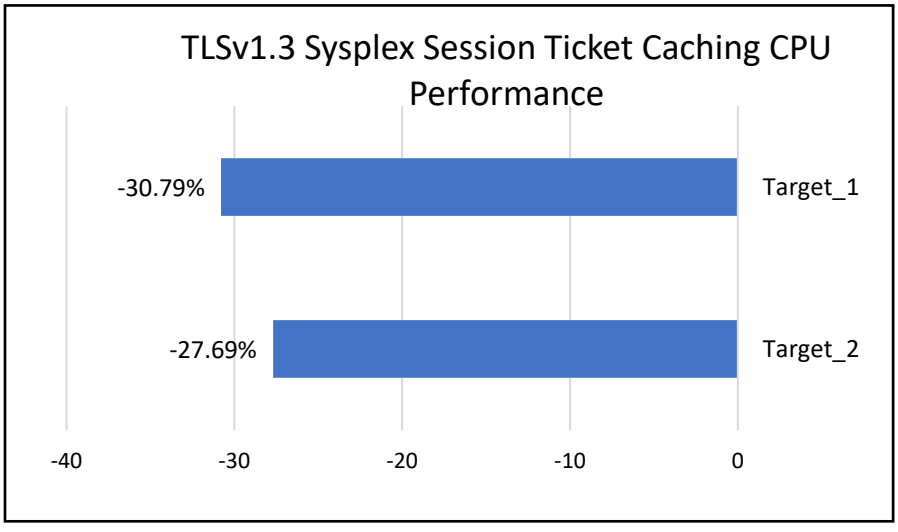
TLSv1.3 Sysplex-wide Session Resumption

AT-TLS (through System SSL) now supports sysplex-wide session resumption for the TLSv1.3 protocol starting in V2R5 [6]. Prior to V2R5, AT-TLS only supported TLSv1.3 session ticket caching within the scope of a single AT-TLS group. The TLS sysplex-wide session ticket caching provides the ability for handshake session ticket information to be shared across multiple systems in a sysplex. The goal of this measurements is to show the performance benefits of enabling this function in a sysplex environment as shown in Figure 8. We see up to **30%** network CPU cost savings on targets when sysplex-wide session tickets caching is enabled for TLSv1.3.



- For these measurements, two targets were used with the System SSL GSKSRVR task running on each target.
- The round robin Sysplex Distributor distribution method was configured to distribute connections.
- The GSKSRVR statistics display showed over 95% of the time, the cached TLS session ticket was used.

Figure 8 Test environment for sysplex-wide session ticket caching



- Chart shows network CPU cost percentage difference relative to **NO** sysplex-wide session ticket caching
- Up to 30% reduction in CPU cost on target systems

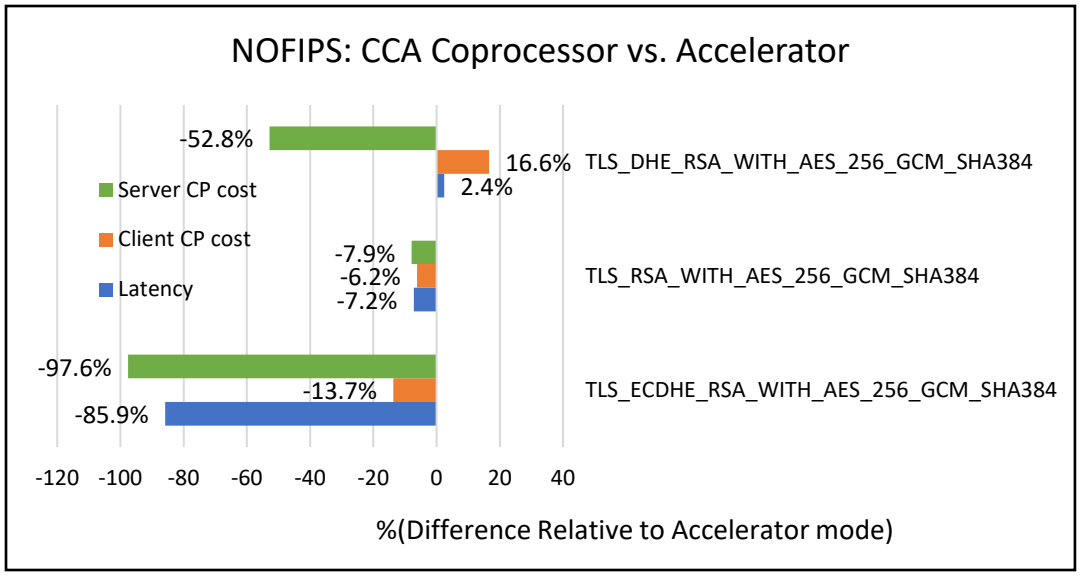
Figure 9 Performance improvements of enabling TLSv1.3 Sysplex Session Ticket Caching

FIPS 140-2 Performance

Federal Information Processing Standard (FIPS) standard 140-2 provides a higher degree of assurance of the integrity of the cryptographic algorithms being used. The goal of this test was to measure the performance impact of enabling FIPS 140-2 [7] support on some TLSv1.2 ciphers.

With FIPS 140-2 disabled, Figure 10 shows different TLSv1.2 protocol cipher suites when running the Crypto Express adapter in Coprocessor mode compared to Accelerator mode. We observed significant performance improvements for the case of the Coprocessor mode, especially for the DHE-based ciphers (up to 98% CPU savings), as shown in the chart.

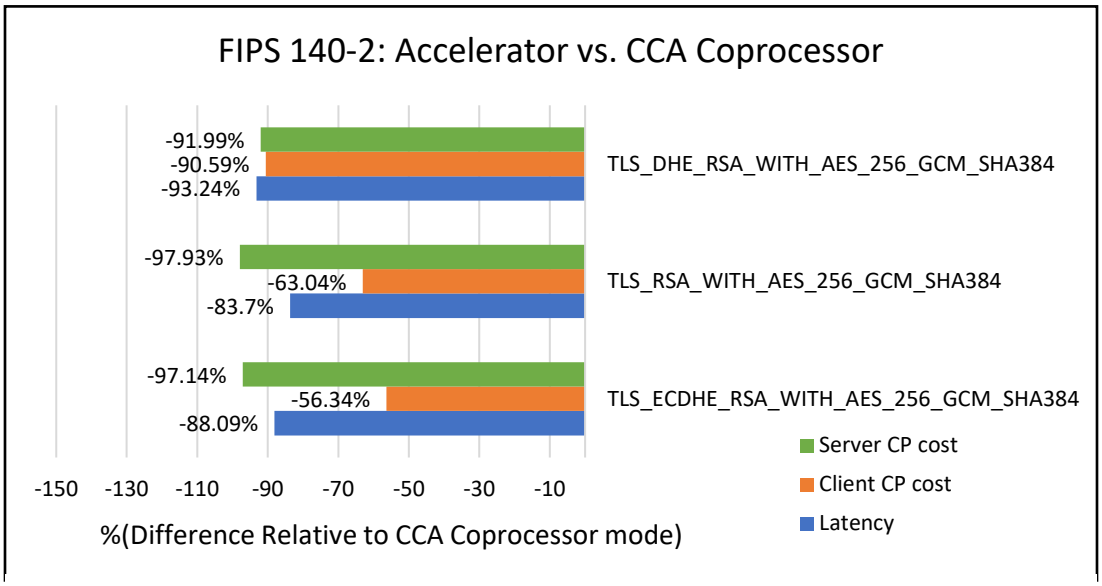
Note: Required RSA signing operations for DHE and ECDHE based cipher suite are offloaded onto the Coprocessor card and not the Accelerator card, which translate into 53% and 98% CPU savings, respectively. There are no signing operations for the TLS_RSA_xxxx based cipher suite, but there are RSA encrypt and decrypt operations. These operations are offloaded to Coprocessor or Accelerator when available.



- Chart summarizes the performance of coprocessor card compared to the accelerator card in non-FIPS mode
- Significant performance benefit in Coprocessor mode compared to Accelerator mode

Figure 10. Performance degradation of using Crypto Express adapter as Accelerator in non-FIPS environment

For FIPS mode execution (Figure 11), System SSL calls ICSF to offload a subset of RSA operations (digital signature generation, verification, encryption, and decryption [2]) to the Crypto Express card when configured in accelerator mode. We observe great CPU cost savings (up to 97%) and 93% reduction in network latency in using Accelerator mode compared to Coprocessor mode in a FIPS environment.



- Chart summarizes the performance of accelerator card in FIPS mode compared to coprocessor card in FIPS mode
- FIPS 140-2 mode using accelerator card results in significant CPU savings and latency reduction

Figure 11. Performance benefits (CPU savings and latency reduction) of using Crypto Express adapter as Accelerator for FIPS environment

Conclusion: *Configuring a mix of your Crypto Express adapters as Coprocessors and Accelerators will ensure optimal performance, whether in FIPS 140-2 mode or not.*

HEAPPOOLS64 Performance

The Language Environment (LE) runtime option, HEAPPOOLS64, is used to control the user heap pool storage management, which can improve performance. The heap pool algorithm virtually eliminates contention for accessing user heap storage in an AT-TLS environment when enabled. AT-TLS under heavy load of concurrent TLSv1.2 handshakes can avoid significant LE latch contention with HEAPPOOLS64. With TLSv1.3, this latch contention appears even with low numbers of handshakes. Regardless of TLS protocol version, the resulting improvements in network performance is impressive. For the purpose of these measurements, we used the default values for the HEAPPOOLS64 runtime option. The results in Figure 12 show the CPU savings and latency reduction of enabling the HEAPPOOLS64 option.

TCP/IP stack wide LE runtime option:

```
//CEEOPTS DD *
HEAPPOOLS64(ON)
```

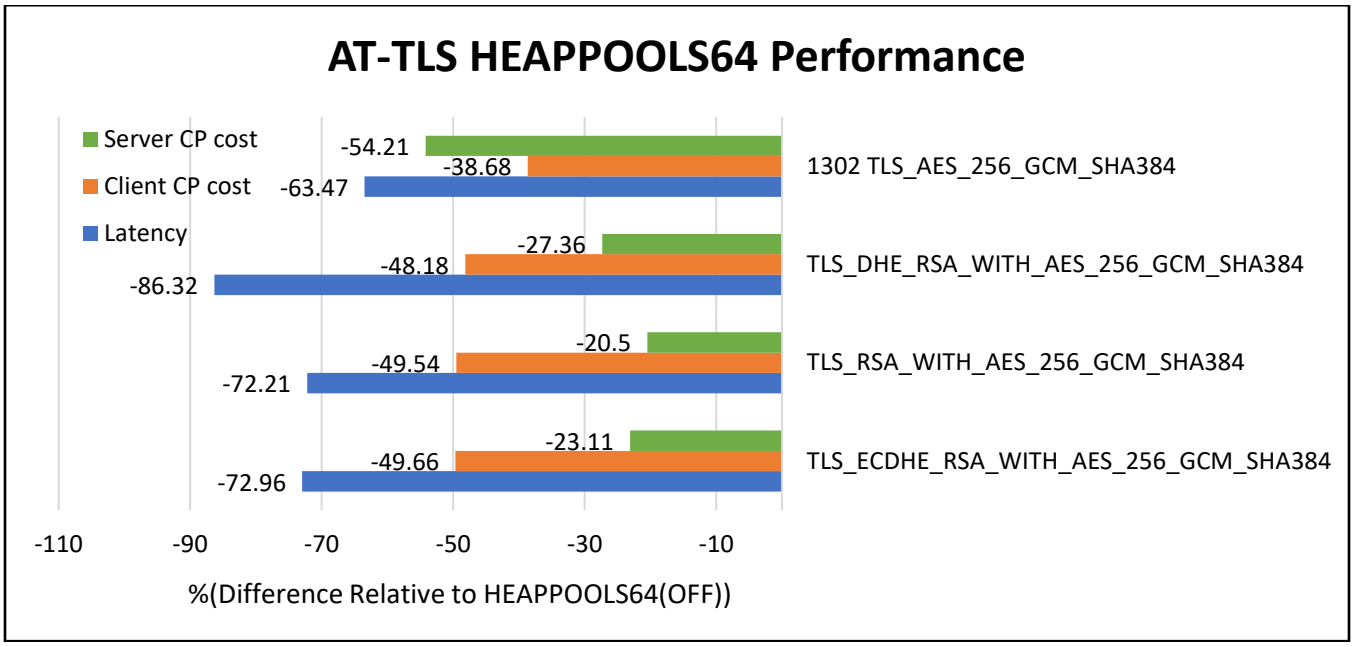


Figure 12. CPU savings and Latency reduction across TLSv1.3 and TLSv1.2 ciphers with HEAPPOOLS64

The performance benefits of enabling LE runtime option, HEAPPOOLS64, is seen across all ciphers. Each cipher in Figure 11 shows tremendous reduction in network CPU and response time. Because of these benefits, users of AT-TLS will now have this runtime option enabled by default.

z/OS Communications Server APAR PH59425 will ensure HEAPPOOLS64 is always enabled.

References

[1]. “IBM Z15 Performance of Cryptographic Operations (Cryptographic Hardware: CPACF,CEX7S”, IBM. Accessed On: May 2023. [Online]. Available: Link

[2]. “z/OS 3.1 Cryptographic Services System Secure Sockets Layer Programming”, IBM. Accessed On: May 2023. [Online]. Available: Link

[3]. “z/OS 3.1 Cryptographic Services Integrated Cryptographic Service Facility Application Programmer’s Guide”, IBM. Accessed On: May 2023. [Online]. Available: Link

[4]. [RFC5246] Rescorla, E., “The Transport Layer Security (TLS) Protocol Version 1.2”, August 2008, https://datatracker.ietf.org/doc/html/rfc5246

[5]. [RFC8446] Rescorla, E., “The Transport Layer Security (TLS) Protocol Version 1.3”, August 2018, https://datatracker.ietf.org/doc/html/rfc8446

[6]. “Sysplex session ticket cache support”, https://www.ibm.com/docs/en/zos/2.5.0?topic=task-sysplex-session-ticket-cache-support

[7]. National Institute of Standards and Technology, FIPS 140-2 Annex D: Approved Key Establishment Techniques, available at URL: Link