CICS Transaction Server for z/OS

Pervasive Encryption

with CICS

Catherine Moxey

IBM CICS

Table of contents

Introduction

How and why should you use Pervasive Encryption with CICS® Transaction Server for z/OS® (CICS)?

IBM® Z Pervasive Encryption provides a big step forwards in helping your organization protect its digital assets and the IBM z14™ offers capabilities that are integrated throughout the stack to provide pervasive encryption.

But if you use, run or manage CICS, you might have a few questions about this:

- Does Pervasive Encryption apply to me?
- What CICS data can I encrypt?
- What is required to use Pervasive Encryption with CICS? How disruptive is it?
- Why would I choose to use Pervasive Encryption with CICS?
- How much does it cost?

If you have any of these questions, then this paper should provide you with answers.

About the author

Catherine Moxey is an IBM Senior Technical Staff Member in the CICS Development team, based in the IBM Laboratory in Hursley in the South of England. Catherine specializes in CICS Analytics and Performance, and also has a focus on CICS File Control and on Resilience.

In a nutshell

You can use Pervasive Encryption with any in-service release of CICS to protect important data in accordance with your organization's policies and without the need to change CICS or your applications.

All the data encryption that is supported by z/OS is also supported for data accessed from CICS, allowing you to encrypt as much or as little of this data as you require.

IBM z14 provides a highly-optimized environment for encrypting your CICS data.

What is Pervasive Encryption, and why is it important?

Data is being described as the 'new perimeter of the enterprise' because probably your most important business asset is your data. If your enterprise's data were to be lost or compromised, the implications would be very serious. Audit requirements often identify a need for encryption, but have been difficult to meet because of the effort involved in changing applications to carry out encryption, the operational impact and overheads, and other factors. Data regulations such as GDPR (General Data Protection Regulation) recommend or require encryption, and can be more easily satisfied by using pervasive encryption. The cost of not encrypting the data in the event of a data breach would far outweigh any overhead involved in encrypting the data.

Pervasive Encryption removes these barriers to encryption to help you both to protect your data and to satisfy audit and compliance requests.

Pervasive Encryption provides a streamlined approach to using encryption to protect that all-important data, allowing encryption to be managed via configuration and administrative controls, rather than requiring application changes.

It makes use of the Central Processor Assist Cryptographic Function (CPACF), which reduces the overhead of the encryption processing, especially on IBM z14.

Pervasive Encryption encompasses the encryption of various kinds of sensitive data, including data set encryption and coupling facility (CF) encryption. It moves away from selective encryption, where the choice of what to encrypt was based on what could be afforded and made more difficult by needing to know exactly where the sensitive data was. It also covers both data-at-rest and data-in-flight. This paper focuses primarily on encryption of data-at-rest, but you can also secure data in flight, coming into and out of CICS, using such capabilities as SSL, AT-TLS, SOAP message encryption and IBM MQ Advanced Message Security (AMS).

Pervasive Encryption brings together capabilities in the hardware, operating system and other components. IBM Z machines provide the hardware infrastructure that makes pervasive encryption possible. Enhancements, specifically in the IBM z14, have reduced the overhead of pervasive encryption, making it an easier choice. The z/OS operating system takes advantage of cryptographic processors to provide data protection for many z/OS data sets, zFS file systems and Coupling Facility structures. z/OS data set encryption includes the data sets that are used by CICS, IMS™, and Db2®. Figure 1 shows how IBM Z provides multiple layers of encryption.

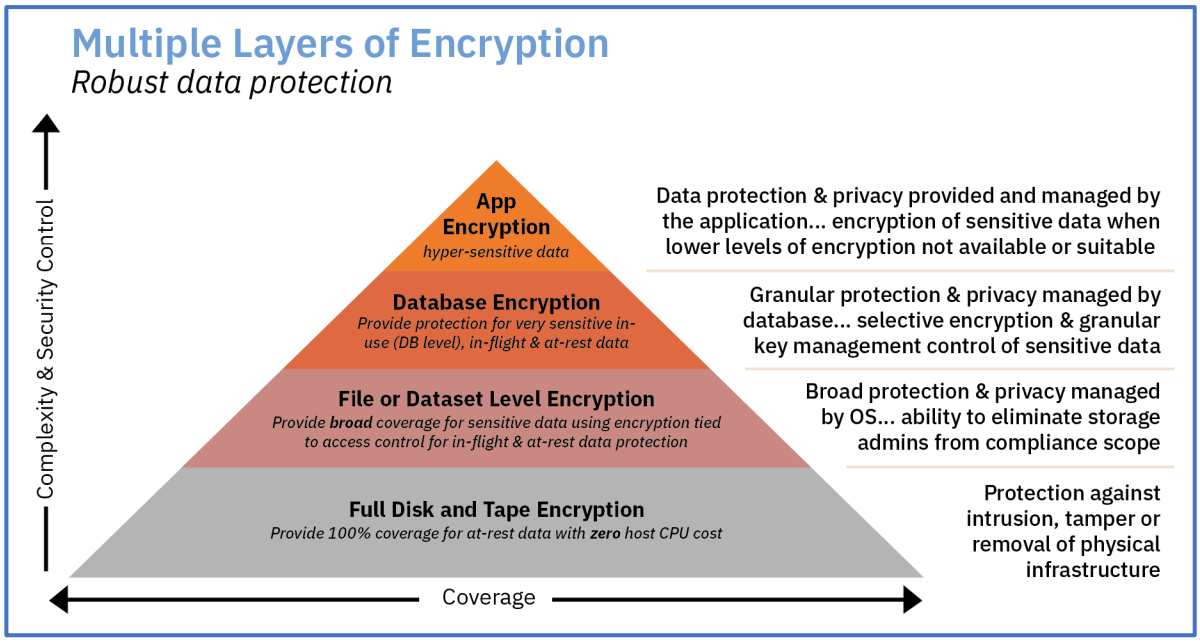


Figure 1: Multiple Layers of Encryption (based on IBM Z14 Pervasive Encryption deck)

What CICS data can be encrypted?

The CICS design point is that anything used by your CICS systems, for which encryption is supported, can be encrypted – aligning with the move away from selective encryption. Pervasive Encryption is transparent to your CICS systems and applications.

You can encrypt any of the data sets that you use with CICS for which z/OS data set encryption is supported. This includes any user data sets (of the supported data set types) that are accessed through CICS File Control APIs, QSAM data sets used for CICS extrapartition transient data, sequential (BSAM) data sets used with CICS, and many of the CICS system data sets.

Although the initial functionality for Pervasive Encryption focused on encryption of data sets, the support also includes other types of data, such as Db2 and IMS data, and, as of z/OS V2.3, coupling facility (CF) data and zFS files.

The CICS data that can be encrypted includes the following:

- User data sets defined to CICS, including key-sequenced data sets (KSDS), entry-sequenced data sets (ESDS), relative record data sets (RRDS), and variable relative record data sets (VRRDS), accessed through base VSAM and through VSAM Record Level Sharing (RLS). Any alternate index associated with a data set can also be encrypted.
- The VSAM KSDS data sets that are used as backing source files for shared data tables or coupling facility data tables
- Db2 tables accessed from CICS
- IMS databases accessed from CICS
- Temporary storage data sets (DFHTEMP)
- Intrapartition Transient Data (DFHINTRA)
- Extrapartition Transient Data
- Auxiliary Trace data sets (DFHAUXT and DFHBUXT)
- CICS dump data sets (DFHDMPA and DFHDMPB)
- Document templates (DOCTEMPLATE resources)
- URIMAPs used for static delivery
- zFS files used for CICS resources such as Bundles and Web services
- CF structures used by CICS for CF data tables, shared Temporary Storage queues, Named Counters, and CICSPlex SM Optimized Workload Management
- BTS repository data sets and BTS local request queue (LRQ) data set
- Global and Local catalog data sets (DFHGCD and DFHLCD)
- CICS system definition data set (DFHCSD)
- CMAC messages data set (DFHCMACD)

This list is in an approximately descending order of suitability for encryption, based on the likelihood of the data containing sensitive information. A table in the CICS Knowledge Center topic on *Planning for data set encryption* gives further information about which data sets are likely to be good candidates for encryption.

How do I choose what to encrypt?

We've shown that CICS provides full support for IBM Z Pervasive Encryption, but how do you decide how much of your data to encrypt? There is a range of scenarios that you might consider.

Focus selectively on sensitive data

One approach is to identify where you have any sensitive data or any personal information, and choose specifically to encrypt that. This might, for example, include the data sets containing information about your customers and their transactions, the CF data table used to check on customer eligibility, as well as temporary storage queues used to keep intermediate data while processing transactions. However, you would not need to encrypt the CICS catalog data sets, as they only contain configuration data about this CICS instance and not any customer-related information. Perhaps you would also decide against needing to encrypt a transient data queue which is just used to trigger time-related processing. This approach would give you protection where it is needed, without incurring even the small overhead associated with encryption where it is not needed.

However, a selective approach requires you to identify, and continue to review, where you have sensitive data.

Encrypt everything

A more straightforward approach would be to encrypt as much of your data as possible, without specific consideration as to its sensitivity, especially if this is needed to satisfy audit requirements that require protection for data regardless of its contents. This avoids any need to review the data, putting such a policy in place would ensure that future data will also be encrypted and that no important data will be missed. In this situation, you would not only encrypt the customer data sets, temporary storage queues and CF data table structures from the previous scenario, but also the catalogs and the transient data queue, and perhaps other resources as well.

Focus on *types* of data that could contain sensitive information

An intermediate approach is also possible, in which you encrypt any *type* of data which has the possibility of containing sensitive or personal information, without the need to review each instance of that type of data. For example, some of the CICS system data sets can potentially contain customer data, most notably transient data and temporary storage data sets, but others only ever contain configuration information, such as the catalog data sets and the CSD.

Regardless of the approach you take, you might decide to start with a few data sets, and roll out encryption further as you become more comfortable with using it. One of the considerations for deciding which data to encrypt could be the expected CPU cost. To help you estimate this, the IBM Z Batch Network Analyzer (zBNA) Tool has support for encryption and can use your SMF data to estimate the overhead based on the characteristics of your production workload. For more details see IBM Z Batch Network Analyzer Tool.

The good news is that all these scenarios are easy to implement and do not even require any changes to CICS or to your applications.

Choosing encryption keys

Another consideration is how many encryption key labels to use, and at what granularity: using only one key label for all your data is probably only a good idea if you are encrypting a small subset, but a key label for every data object would be unwieldy and difficult to manage. There is a range of possibilities, but some people opt for a key label per business application, while others segment things at a more technical level. You might want to consider the naming convention for your encryption keys, and how this should relate to your desired granularity.

You will also want to think about and plan your key management policy. The policy you want to adopt will need to be determined by your organization, but IBM offers several tools to help you manage keys. These include ICSF, the Trusted Key Entry (TKE) Workstation, and IBM Enterprise Key Management Foundation (EKMF). For further information, see *z14 Introduction to Enterprise Key Management* in the IBM Crypto Education Community.

How do I set up encryption?

So, after you have decided what CICS data to encrypt, how do you go about doing this? For full details, you should refer to the z/OS documentation, but this section summarizes the steps involved. Many of the steps are one-time activities when you first start out on your Pervasive Encryption journey.

There are some things that you will want to decide in advance before carrying out these steps. For example, who in your organization will be responsible for assigning keys? What mechanism will be used to assign key labels and what granularity of key labels do you want to have? Who will have access to create encrypted data sets?

Figure 3 at the end of this section is a flow chart which highlights the main steps involved in setting up encryption for each of the types of data covered here.

Setting up Data Set Encryption

The steps involved in setting up data set encryption are:

1) Ensure that you have the necessary, minimum pre-requisites:
    - IBM z196 or later hardware.
    - z/OS 2.3, or z/OS 2.2 with the PTF for APAR OA50569 applied.
    - Crypto Express3 Coprocessor or later.
    - Feature 3863: CP Assist for Cryptographic Functions (CPACF).
    - z/OS Cryptographic Services Integrated Cryptographic Service Facility (ICSF) installed and configured.

You need at least the above pre-requisite levels, but the optimum hardware configuration is IBM z14 and the Crypto Express6s Coprocessor.

2) Set up one or more encryption keys, and key labels to identify them, in the ICSF key repository (CKDS). You can find details on how to do this in the z/OS documentation in *Managing Cryptographic Keys Using the Key Generator Utility Program*.
3) Grant access to encrypt data sets: Define the required authorities using RACF or equivalent security product, to make data set encryption available to users who are authorized to use it, and also importantly make it unavailable to those who are not. The following is a high-level summary; for further details see *Protecting data sets for data set encryption*
    - The STGADMIN.SMS.ALLOW.DATASET.ENCRYPT profile in the FACILITY class is used to control access to create encrypted data sets: at least READ authority to this resource is required to create encrypted data sets, except when the key label is specified in the DFP segment in the RACF data set profile.
    - The CSFKEYS general resource class is used to control access to key labels.
    - The CSFSERV general resource class (resource CSFKRR2) might additionally be needed to control access to cryptographic services, but only if CHECKAUTH(YES) is specified on the ICSF installation options data set.

    You are now ready to encrypt your first data set
4) Make a copy of each data set to be encrypted. Ensure that the copy specifies Extended Format, which is required for encryption. Specify the key label of the key to be used to encrypt the data in one of the following ways:
    - On the RACF data set profile, using the DATAKEY keyword.
    - In JCL, using DSKEYLBL.
    - On TSO allocate, using DSKEYLBL.
    - On dynamic allocation, through the DALDKYL text unit.
    - On IDCAMS DEFINE, using the KEYLABEL parameter of DEFINE CLUSTER. Any alternate index associated with the CLUSTER will also be encrypted using the same key label.
    - On the SMS data class, specifying the Data Set Key Label field on the DEFINE/ALTER panel.

    If more than one of these is specified, then the order of precedence is (i) RACF data set profile, (ii) JCL, dynamic allocation, TSO allocate, or IDCAMS DEFINE, and (iii) SMS data class.

    Figure 2 shows a JCL snippet to define a data set `PAYROLL.SIMPLE.FILE` as encrypted, by using dataclass `EXTCLAS` which specifies Extended Format and using a keylabel `LABEL.FOR.PAYROLL` for the encryption.
5) Copy data across from the old data set to the new encrypted data set, using a function such as REPRO.
6) You will probably want to delete the original data set and rename the new, encrypted data set to the original name.

```
//************************************************************
//*** This defines an encrypted VSAM file, using data class  ***
//*** EXTCLAS, and a KEYLABEL that was defined via ICSF.     ***
//*** EXTCLAS was defined with a DSN Type of EXTENDED (and   ***
```

```
//*** the Extended Addressability attribute set to NO).      ***
//************************************************************
DEFINE CLUSTER(NAME(PAYROLL.SIMPLE.FILE) -
               . . . -
               DATACLASS ( EXTCLAS ) -
               KEYLABEL ( LABEL.FOR.PAYROLL ) -
               )
```

Figure 2: Simple example showing defining an encrypted data set

Setting up Coupling Facility Encryption

Coupling Facility data can be encrypted on systems at z/OS V2.3 and above, running on IBM zEnterprise EC12 or BC12 servers or newer. List and cache structure entry and entry adjunct data can be encrypted while the data is being transferred to and from the coupling facility and while the data resides in the coupling facility structure. Encryption of structure data can be controlled on a structure-by-structure basis.

If not all systems in the sysplex support connecting to encrypted structures; that is, some systems are at z/OS V2.2 or below, coexistence APAR OA52060 must be installed on the down-level systems.

Each structure definition in a CFRM CDS that specifies ENCRYPT(YES) requires an encryption key. The administrative data utility for CFRM must be run to assign an initial encryption key to a structure definition. The SETXCF MODIFY command can be used to change a structure's encryption key in the active CFRM policy.

For further details, including how to set up the authorization needed to encrypt CF data, refer to *Encrypting coupling facility structure data* in the *z/OS MVS Setting up a Sysplex* information in the IBM Knowledge Center.

Setting up zFS Encryption

Encrypting of z/OS File System (zFS) files requires z/OS 2.3, and all systems in a sysplex must be at z/OS V2.3 or later before encryption can begin. The encryption setting applies to the entire file system, and is not controlled on a file-by-file basis.

New zFS file system data can be encrypted, with the file system being defined and formatted so that any data added to the zFS file system is automatically encrypted. You can use `format_encryption=on` in your IOEFSPRM configuration file if you want data in all new zFS file systems to be automatically encrypted.

Existing zFS file system data can also be encrypted. Encrypting an existing file system is a long-running administrative command and application access is fully allowed to the file system during the operation.

The encryption process uses DFSMS encryption support. When zFS encrypts a file system, it encrypts all security information, access control lists, symbolic link contents, and file contents. The RACF userid associated with the zFS address space will need to be given access to any key labels, through the CSFKEYS resource class.

For further information and considerations about encrypting zFS, refer to the z/OS V2.3 *z/OS Distributed File Service zFS Administration*.

Setting up Db2 Encryption

Db2 data that supports encryption can be used with CICS. Db2 for z/OS uses z/OS DFSMS data set encryption to transparently encrypt Db2 data sets, such as those associated with a particular table. Db2 for z/OS does not provide database administrator controls for encryption. You enable and manage data set encryption by using DFSMS data set encryption as described above.

The pre-requisites for Db2 encryption are:

- As for data set encryption: IBM z196 or later hardware, Crypto Express3 Coprocessor or later, Feature 3863: CP Assist for Cryptographic Functions (CPACF), and z/OS Cryptographic Services Integrated Cryptographic Service Facility (ICSF) installed and configured.
- z/OS 2.3, or z/OS 2.2 with the PTFs for APARs OA50569 and OA53951 applied.
- Db2 11 using z/OS new function mode, and with APAR PI81900, or Db2 12 with APAR PI81907 (for M100 level).
- The Db2 started task user ID and any user ID that is required to read or write to an encrypted data set is permitted to use any key labels that are used to protect Db2 data sets.
- Any key label that is used to protect Db2 data sets is defined on all the members of a data sharing group and on any backup systems that might read or write from an encrypted data set. With Db2 12, you can define key labels at the Db2 object level or the RACF data set profile level.
- Any user ID that is required to run any of the stand-alone utilities is authorized to use any key label that is used to protect Db2 data sets.

If you use any third-party Db2 tools, utilities and add-on products, then check with your vendor for any required support.

Db2 is designed to transparently encrypt data at rest without database downtime or requiring the administrator to redefine objects. This includes the ability to transparently encrypt Db2 logs, catalog, directory, tables and indices including all data types such as large binary objects. Refer to the Db2 for z/OS documentation for further details: *Encrypting your data with z/OS DFSMS data set encryption*.

Setting up IMS database encryption

All the IMS databases that support pervasive encryption can be used with CICS, and accessed via IMS DBCTL. IMS Fast Path (DEDB) databases and IMS Full Function VSAM databases both support encryption.

IMS 13 and later releases support data set encryption. The z/OS and hardware prerequisites for encrypting IMS data sets are the same as for Data Set Encryption.

IMS data sets that are to be encrypted must be defined as SMS-managed extended format data sets with a key label associated them. The key label can be specified through one of the following methods:

- Create SAF rules that associate a key label with a data set name pattern by using the DATAKEY parameter of the DFP RACF segment.
- Specify a key label by using JCL, dynamic allocation, or TSO allocate (DSKEYLBL parameter).
- Specify a key label on the IDCAMS DEFINE command (KEYLABEL parameter).
- Use the DATACLAS parameter with a key label that is associated with it.

For further information, refer to *Data set encryption support for IMS*.

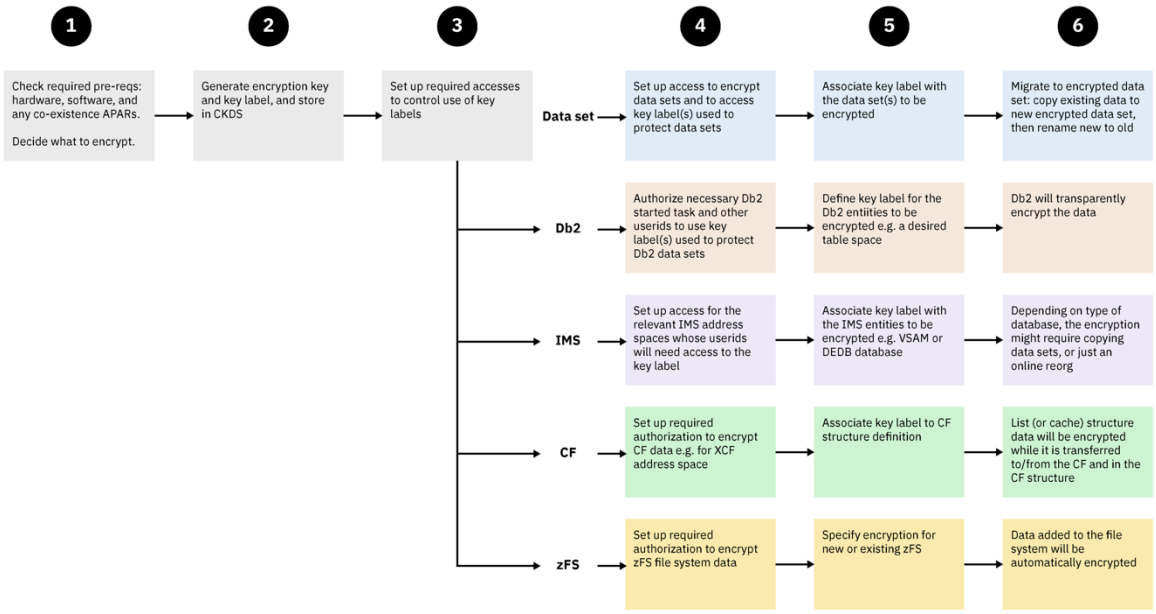


Figure 3: Flow chart for setting up encryption for different types of CICS data. (If you're printing this paper, see Flow chart for setting up encryption for different types of CICS data (landscape version)

What is the performance impact of pervasive encryption with CICS?

With the right levels of software, encryption can be used on any IBM Z machine that supports that software. However, enhancements introduced in the IBM z14 machine make pervasive encryption a natural fit due to a specific focus on optimizing the encryption. For example, according to a Solitaire Interglobal report, *"on-chip cryptographic acceleration was enhanced to provide more than 6x more performance than z13 at more than 18x faster than competitive platforms."* (Source: *IBM Z Pervasive Encryption Marks a Paradigm Shift for Security* in IBM Systems Journal.)

This section describes internal IBM benchmarks that were used to measure the overhead introduced by encrypting CICS data on an IBM z14 in different scenarios. These are included as examples to show the low overhead of encryption on an IBM z14, but as mentioned earlier, you can use the zBNA tool to estimate data set and zFS encryption overheads for your production workload.

The first benchmark compared a CICS-VSAM workload where no data had been encrypted with the same workload in which all customer-sensitive data located either on DASD or in CF structures was encrypted. This could be regarded as using the 'encrypt everything' approach described earlier. The second benchmark compared a similar CICS-VSAM workload, but in this case the comparison was between no encryption and encryption of customer-sensitive data on DASD but not in the CF. This is similar to the 'selective encryption' approach.

Figure 4 shows a schematic of the workload environments used in these two measurements, to illustrate which data was encrypted in the two cases.

A third benchmark looked at the overhead of zFS encryption.

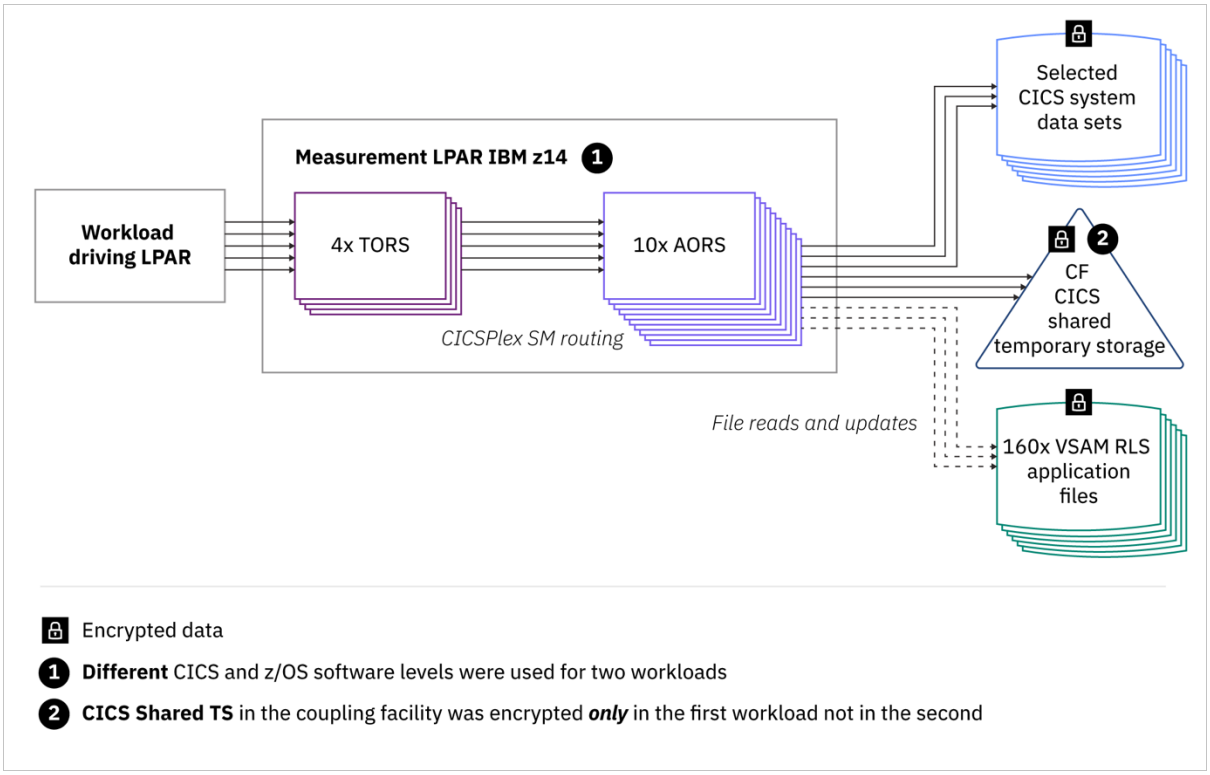


Figure 4: CICS-VSAM workload used for the performance measurements

The first internal IBM CICS benchmark was measured using two dedicated LPARs on an IBM z14, configured as part of a sysplex, and running on z/OS V2.3 with CICS TS V5.2. An internal coupling facility (CF) was co-located on the same central processor complex as the measurement and the driving LPARs, connected through internal coupling peer links. An IBM System Storage® DS8800 unit was used to provide external storage.

The workload used was a non-threadsafe COBOL application that accesses VSAM RLS files as well as CICS shared temporary storage. **All** customer sensitive data located either on DASD or CF structures was encrypted.

The measurement showed that in this scenario, **enabling dataset encryption and CF encryption increased the CPU cost per transaction by only 4%.**

The second internal benchmark used a similar non-threadsafe COBOL application which also accessed VSAM files, and was configured to accept work into four Terminal Owning Regions (TORs), which dynamically routed transactions using CICSPlex SM workload manager to ten Application Owning Regions (AORs). The AORs each accessed VSAM files using RLS, and all regions were connected using MRO/XM connections. The workload used CICS TS V5.4 and z/OS V2.2, and the measurement LPAR was configured with 16 dedicated central processors and 16GB of real storage. The workload compared running with no encryption and running with encryption enabled for all customer sensitive data residing on DASD (but not in CF). All 160 VSAM application files as well as the CICS system data sets DFHTEMP, DFHINTRA, DFHLRQ, DFHLCD, and DFHGCD were configured to use data set encryption. In this case, the data stored in temporary storage queues in the coupling facility was not encrypted.

In this scenario, the workload showed an overhead of **only 1.6% CPU per transaction** when data sets containing sensitive data were encrypted compared with when they were not encrypted. You will notice that different software levels were used in the two measurements, so the results are not directly comparable between the two.

A third internal measurement compared the performance of a CICS workload using encrypted zFS files with the same workload using unencrypted zFS files. This used a CICS Liberty servlet with z/OS V2.3 and a development level of the CICS TS V5.5 open beta. When running at a steady state, this workload created a small amount of zFS file activity through Liberty lookup and logging activity, and zFS file activity was greatly increased by activating CICS tracing for the component associated with Java Liberty processing. The CPU costs of running with and without tracing using encrypted and unencrypted zFS files were measured. **The additional cost of using encrypted zFS was observable but very small**: for the workload investigated it added less than 1% to the total CPU cost.

Disclaimer: *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. All measurements were run multiple times, and the results were averaged.*

A few additional considerations

This includes a few other things to consider when using pervasive encryption with CICS.

Extended Format is required to encrypt data sets, but it is easy to inadvertently specify Extended Addressing at the same time as Extended Format, which might not be what you want. Not all CICS System data sets support Extended Addressing, most notably temporary storage, intrapartition transient data, and dump data sets, and unless your application exploits Extended Addressing, you will not want to specify this for your user data sets.

CICS extrapartition transient data can use a partitioned data set (PDS), but PDS encryption is not supported. This also means that program library data sets used for CICS programs, whether PDS or PDSE (partitioned data set extended) format, cannot be encrypted.

Encrypted data sets can also be compressed. If you use compression, then the data should be compressed before it is encrypted. By using compressed-format data sets, the access method will ensure that the compression and encryption are carried out in the correct order.

If you encrypt trace or dump data sets, you will need to decrypt these before sending them to IBM for diagnosis.

It is worth noting that if you are using encryption at z/OS 2.2 and upgrade to z/OS 2.3, there are no changes needed to any of your encryption set up after the upgrade. z/OS 2.3 incorporates all the support required for encryption, and existing encrypted data sets will just continue to work as before.

The CICS Knowledge Center provides information about Encrypting data sets and Planning for data set encryption (see Find more information). This information is only included in the CICS TS V5.4 Knowledge Center and later releases, but it applies equally to all in-service releases of CICS TS.

Summary and Conclusions

We hope this article has shown you that Pervasive Encryption does indeed to apply to you, that you can encrypt any CICS data for which encryption is supported, that after a few one-time set up steps it is very easy to use Pervasive Encryption with CICS, and that it can make good business sense to use Pervasive Encryption with CICS.

Find more information

Here is your one-stop-shop for further reading around the topic.

For an introduction to Pervasive Encryption:

- 'Is your enterprise encryption strategy a compromise?' – series of blog posts by Mike Jordan, starting at https://www.ibm.com/blogs/systems/enterprise-encryption-strategy-compromise/
- 'Enabling pervasive encryption through IBM Z stack innovations' in IBM Journal of Research and Development (Volume 62, Issue: 2/3, March-May 1 2018), Page(s): 2:1 - 2:11
- http://ibmsystemsmag.com/mainframe/trends/security/enterprise-encryption/

For more on CICS and data set encryption, see topics in the CICS Knowledge Center, in the Configuring information:

- Encrypting data sets
- Planning for data set encryption

For data set encryption, see the Redbook: 'Getting Started with z/OS Data Set Encryption'.

Topic and subtopics on 'Data Set Encryption' in z/OS DFSMS Using Data Sets:
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.idad400/ugencryption.htm

CF encryption details in z/OS MVS Setting Up a Sysplex:
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.ieaf100/encryptstrdat.htm

zFS encryption in zFS administration guide:
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.ioea700/encryptcomp_info.htm
then
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.ioea700/encryptprocess.htm
and links for

- Creating a new file system that is always encrypted on DASD
- Encrypting existing file system data

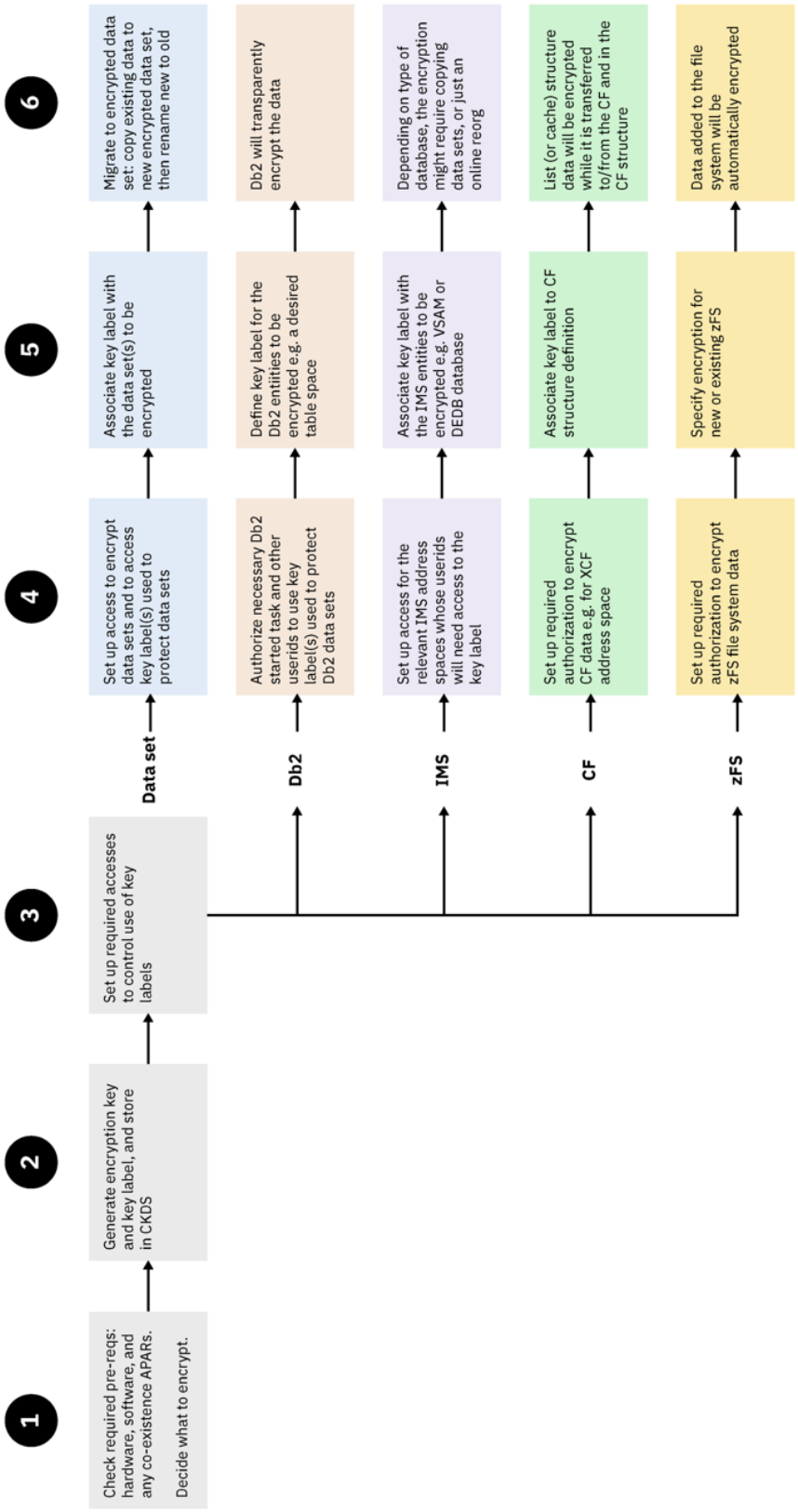
Db2 encryption introductory blog post:
https://www.ibm.com/developerworks/community/blogs/897a7c98-57af-4523-9cfa-07ebc3f996b4/entry/Db2_11_for_z_OS_support_for_z_OS_data_set_encryption?lang=en

Data set encryption support for IMS
https://www.ibm.com/support/knowledgecenter/en/SSEPH2_15.1.0/com.ibm.ims15.doc.sag/system_admin/ims_dataset_encryption.htm

Estimating the effect of encryption using the IBM Z Batch Network Analyzer (zBNA) Tool:
IBM Z Batch Network Analyzer Tool.

IBM Crypto Education Community
https://www.ibm.com/developerworks/community/groups/community/crypto

Flow chart for setting up encryption for different types of CICS data (landscape version)



Notices

Notices applicable to this publication are available here
https://developer.ibm.com/cics/legal-notices/

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

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.