zEnterprise Data Compression

Frank Kyne (frank@watsonwalker.com)

Introduction

- Thank you all for coming.
- And thanks to Jerry and New Era for asking me along. They do a great job and provide a vital service to the z community.
- These slides are a subset of one part of the ITSO 2015 Performance and Availability day, so thank you to ITSO as well for letting me use these slides.
  - And refer to the zEDC Redbook and Redpaper for more info
- Who am I?
- Please ask questions as I go along.

What we are going to talk about

- Understanding compression
- Quick introduction to zEDC
- Potential benefits of zEDC
- List of current IBM and ISV exploiters for zEDC
- Performance comparisons to traditional DFSMS compression options
- Identifying the potential value of zEDC for you
- Configuring zEDC
- How to enable zEDC exploitation by SMF and DFSMS
- Implementation tips
- Hardware and Software prereqs for zEDC
- Summary
- Reference information

Understanding compression

- From a z Systems perspective, there are two main types of compression:
  - Traditional compression.
    - Software – CSRCESRV. (Run Length Encoding) Replaces runs of a character with a smaller number of bytes indicating the count and the character.
    - Hardware – CSRCMPSC – (Dictionary-based) Replaces common character strings with a shorter identifier of the string. This uses the CMPSC hardware instruction.
    - These are both designed to have a low cost of decompression on the basis that you will compress the data once, but decompress many times.
  - "New" zEDC compression. The algorithm used by zEDC compresses data by replacing characters with pointers to identical strings earlier in the block. **So the larger the block of data being compressed, the more effective this form of compression can be**. Files compressed using zlib (which can use zEDC) can be shipped to other platforms and be decompressed using common tools.
- In relation to data set compression on z/OS, the terms 'compression' (in IGDSMSxx member) and 'compaction' (in Data Class definition) are used interchangeably.

Introduction to zEDC

- zEDC is a combination of specialized PCIe cards that deliver very high performance compression and decompression services, software services to communicate with the card, and exploitation in various software products. It consists of:
  - Hardware
    - Up to 8 PCIe adapters per CPC, max of 2 per PCIe drawer domain
    - Up to 15 LPARs sharing each adapter
      - IBM recommend a minimum of 4 adapters
    - And don't forget your Disaster Recovery CPCs!
  - Software
    - There is a zEDC chargeable feature for z/OS that must be licensed on every CPC that will use zEDC.

Introduction to zEDC

- A very important point that you must understand to avoid confusion about zEDC is that it is a system service that will compress or decompress data for *anyone*. For example:
  - The data owner might compress the data. As an example, SMF itself calls zEDC to compress the data before it is sent to the SMF log stream. As far as System Logger is concerned, it is being sent 1s and 0s. It is irrelevant to it whether the data was compressed before it was given to it.
  - Or, the access method might use zEDC to compress the data. In that case, the user program generally has no idea that the data is going to be compressed – it just passes the data to BSAM or QSAM. The access method then takes that data, passes it to zEDC, and writes the compressed data to disk.
  - Or, you could have something like DFSMSdss, which can (optionally) call zEDC itself, OR it can pass the data to BSAM or QSAM which can call zEDC, OR, it could call zEDC *and* pass the compressed data to BSAM or QSAM which might call zEDC again.

Introduction to zEDC

- If someone says that VSAM doesn't support zEDC, that means that VSAM itself will not call zEDC (you *can* still compress VSAM KSDSs using *traditional* DFSMS compression).
- However VSAM data sets could happily contain data that had been compressed by zEDC before it was passed to VSAM, as is the case with System Logger offload data sets for compressed SMF log streams.

Potential zEDC benefits

Based on IBM tests and user experiences, IBM expects zEDC to achieve (on average) about double the compression of traditional z/OS compression techniques. This means **SAVINGS ON DASD SPACE**, reduced load on disk cache, ports, channels, switches, etc.

- And that is if you *already* use compression today! If, like many customers, you *don't* use compression because of the CPU cost, zEDC will **save you even more DASD SPACE** in return for a small increase is CPU time.

If you compress HSM and/or DSS data that is written to tape today, using zEDC instead of traditional DSS or HSM compression **can reduce the number of tapes** you are using and also reduce CPU time.

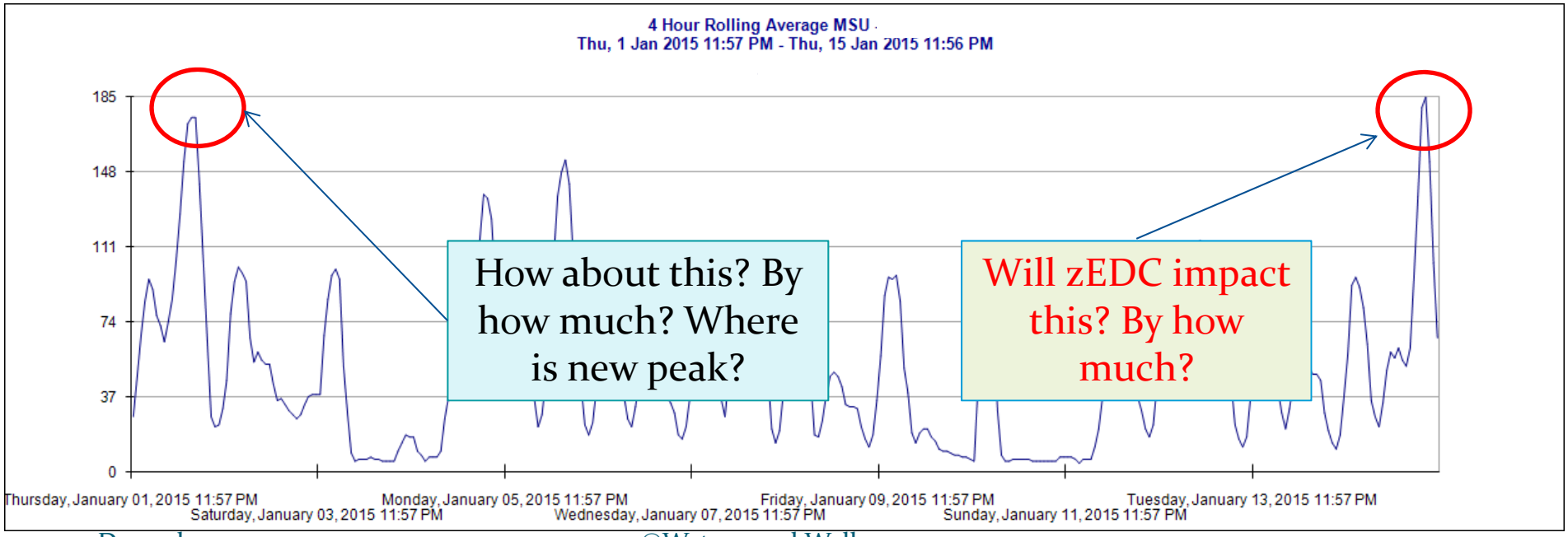
- If you are using a virtual tape subsystem that compresses all incoming data, zEDC will reduce the volume of data being sent through the I/O subsystem and the number of z/OS I/Os.
  - Plus, in some cases, compressing the data twice (once with zEDC and once by the tape subsystem) can result in smaller files than if it was only compressed once.

Potential zEDC benefits

- Because zEDC moves the load of compressing and decompressing data from the CPs to the PCIe card, zEDC **FREES UP z/OS CPU CAPACITY** that was being used to compress data sets.
  - One zEDC card costs USD12K. How much does one CP cost?
- If you encrypt the data, compressing it first reduces the amount of data to be encrypted/decrypted, thereby reducing elapsed time and CPU utilization.
  - Additionally, PTF UA72250 enhances the Encryption Facility for z/OS to support zEnterprise Data Compression (zEDC) for OpenPGP messages.

Potential zEDC benefits

- Because the capacity used on the zEDC card does not factor into your software bill, it *might* result in **REDUCED SOFTWARE BILLS.**
  - This depends on whether you were using compression previously, and if the times when a lot of compression/decompression was being done coincided with your peak R4HA.
  - Also need to make sure that the capacity freed up by zEDC is not eaten by other work – if it is, your SW bills will not reduce.



List of zEDC Exploiters

- The following IBM products provide exploitation of zEDC at this time:
  - SMF (only in logstream mode)
  - BSAM and QSAM (but not VSAM KSDSs)
  - DFSMSdss
  - DFSMShsm when using DFSMSdss as the data mover
  - IBM Encryption Facility
  - MQ Series V8
  - Connect Direct
  - Zlib services
  - Java V7
  - Content Manager OnDemand, starting in 9.5.0.3 (shipped August 14, 2015)
- In all cases, zEDC exploitation is optional, so you must do something to explicitly indicate that you want to use it.

List of zEDC Exploiters

- The following ISV products provide exploitation of zEDC at this time:
  - Data21
  - Innovation Data Processing
  - PKWARE - PKZIP and SecureZip
  - Alebra - Parallel Data Mover
  - Software AG - Entire Net-Work
  - Gzip 1.6 in Rocket Software Ported Tools for z/OS – see http://www.rocketsoftware.com/free-tools

  - If you know of any others, please let us know.

Quantifying zEDC benefits – comparison to traditional compression

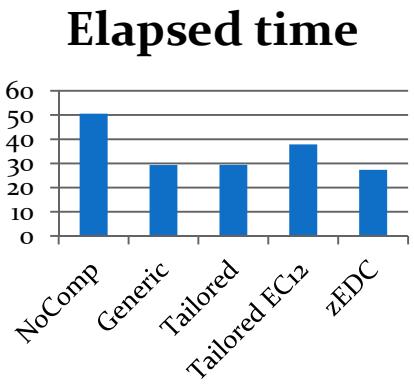
- The most common IBM exploiters of zEDC compression at this time are:
  - Sequential data sets (accessed using BSAM or QSAM)
  - SMF Log streams
  - DFSMSdss and hsm

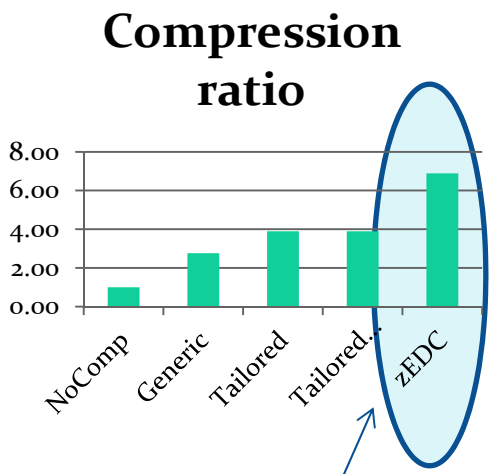
- We ran some comparisons of different types of compression (DFSMS Generic compression, DFSMS Tailored compression (on both zEC12 and z13), and zEDC compression) on 3 different types of data:
  - SMF Sequential data sets.
  - DB2 archive log offloads
  - SVC dump data sets
- We also ran some comparisons of decompression for the different compression methods.
- And we have some measurements for dss DUMP processing

Performance comparison

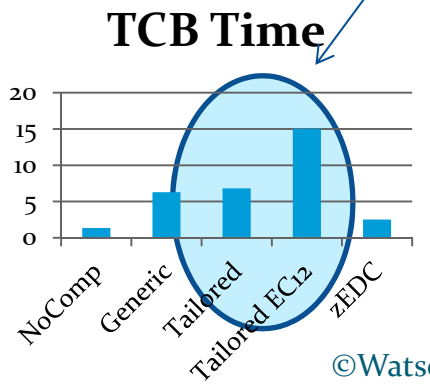
Customer experience for SMF compression ratio has been in the 8-10x range, depending on the SMF record types

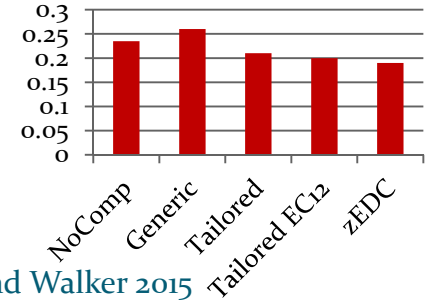
- Measurement 1 - SMF.
- 2 data sets containing SMF data, total size 60255 tracks.
- All runs on z13, except Tailored_EC12, which ran on zEC12.





Compression improvements in z13





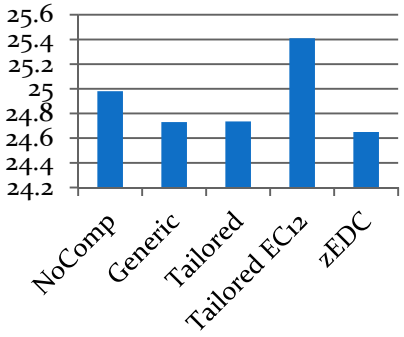
About double

Performance comparison

- Measurement 2 – DB2 logs.
- 15 DB2 archive log data sets, total size 32175 tracks.
- All runs on z13, except Tailored_EC12, which ran on zEC12.

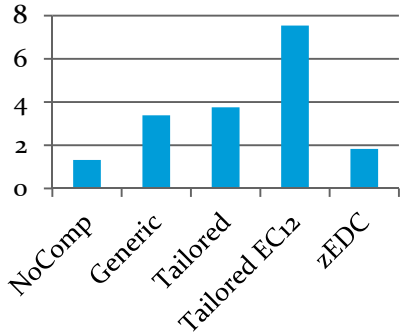
Consistently around 2x compression of traditional compression

Elapsed time

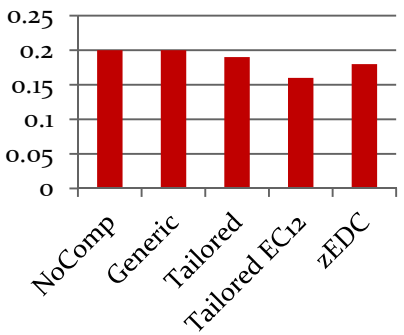


Typically little change in SRB time

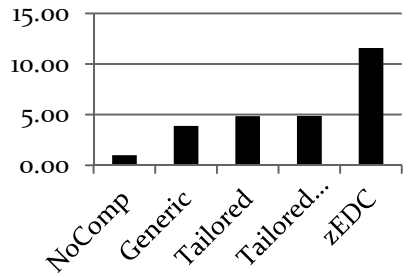
TCB Time



SRB Time



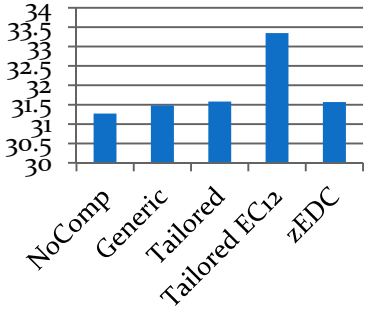
Compression ratio



Performance comparison

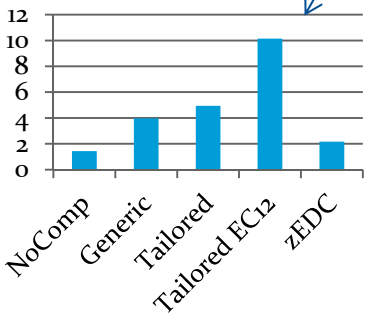
- Measurement 3 – SVC dumps.
- 5 SVC dump data sets, total size 40883 tracks.
- All runs on z13, except Tailored_EC12, which ran on zEC12.

Elapsed time

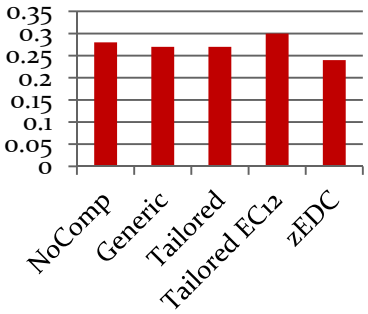


z13 reduces traditional compression CPU time by up to 50% for both tailored and generic

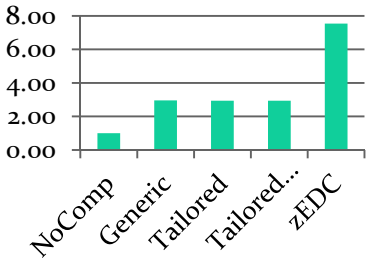
TCB Time



SRB Time

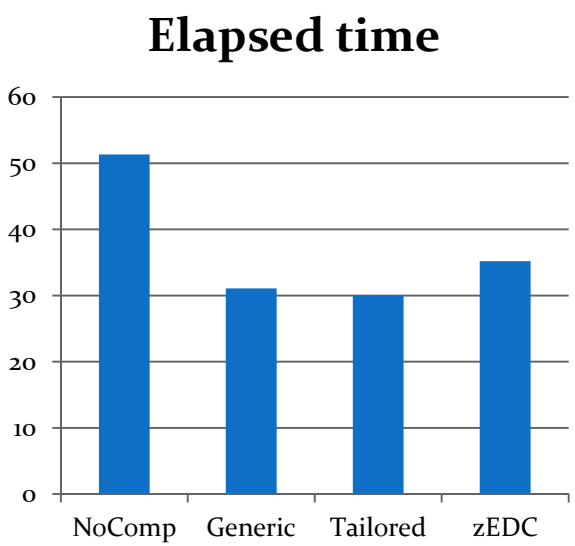


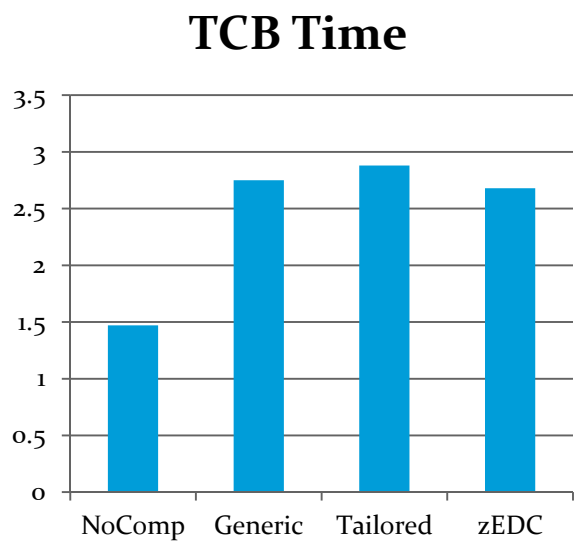
Compression ratio

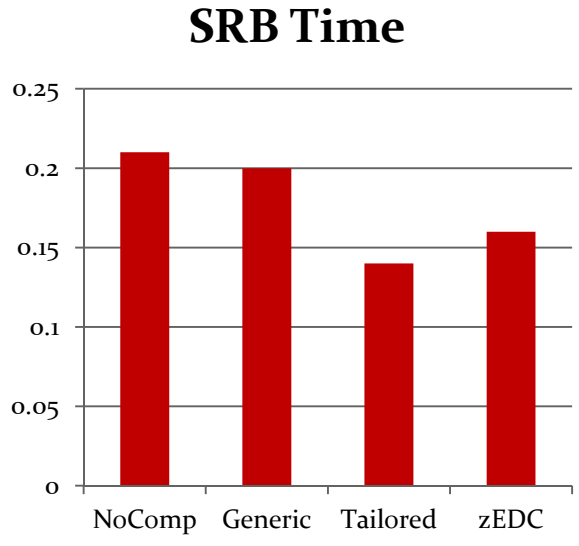


Performance comparison

- Measurement 4 - decompression.
- Decompression comparisons – uncompressed data set size 60255 tracks.
- Measured on both zEC12 and z13.







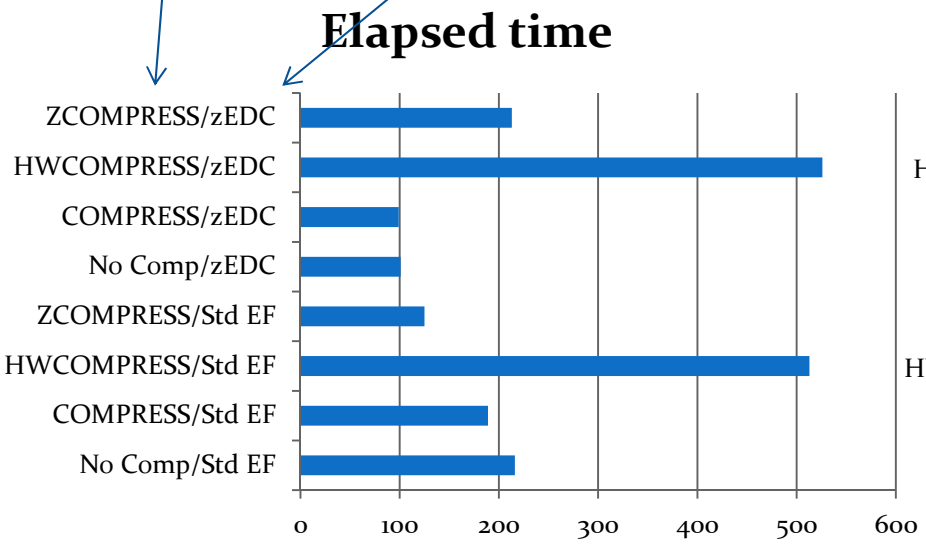
- This is for SMF data set on zEC12. Similar pattern for all data types across both CPCs.

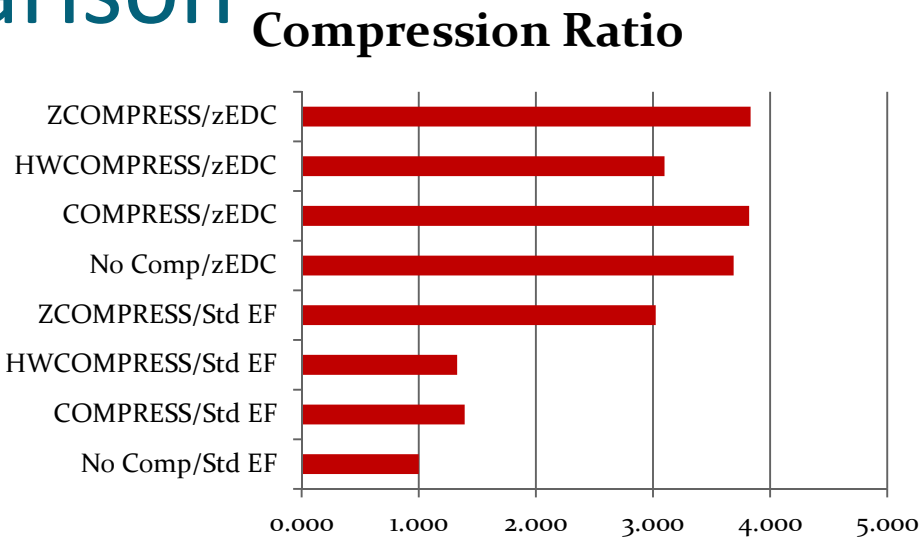
Performance comparison

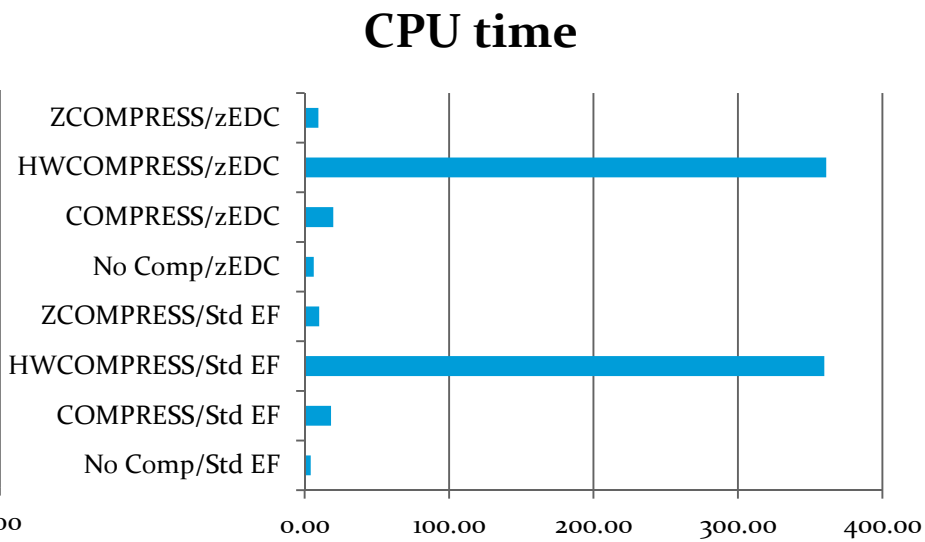
- Measurement 5 – DFSMSdss DUMP.
- DFSMSdss FULL VOLUME – (on zEC12).

DFSMSdss compression option

Output data set format







Identifying potential benefit of zEDC

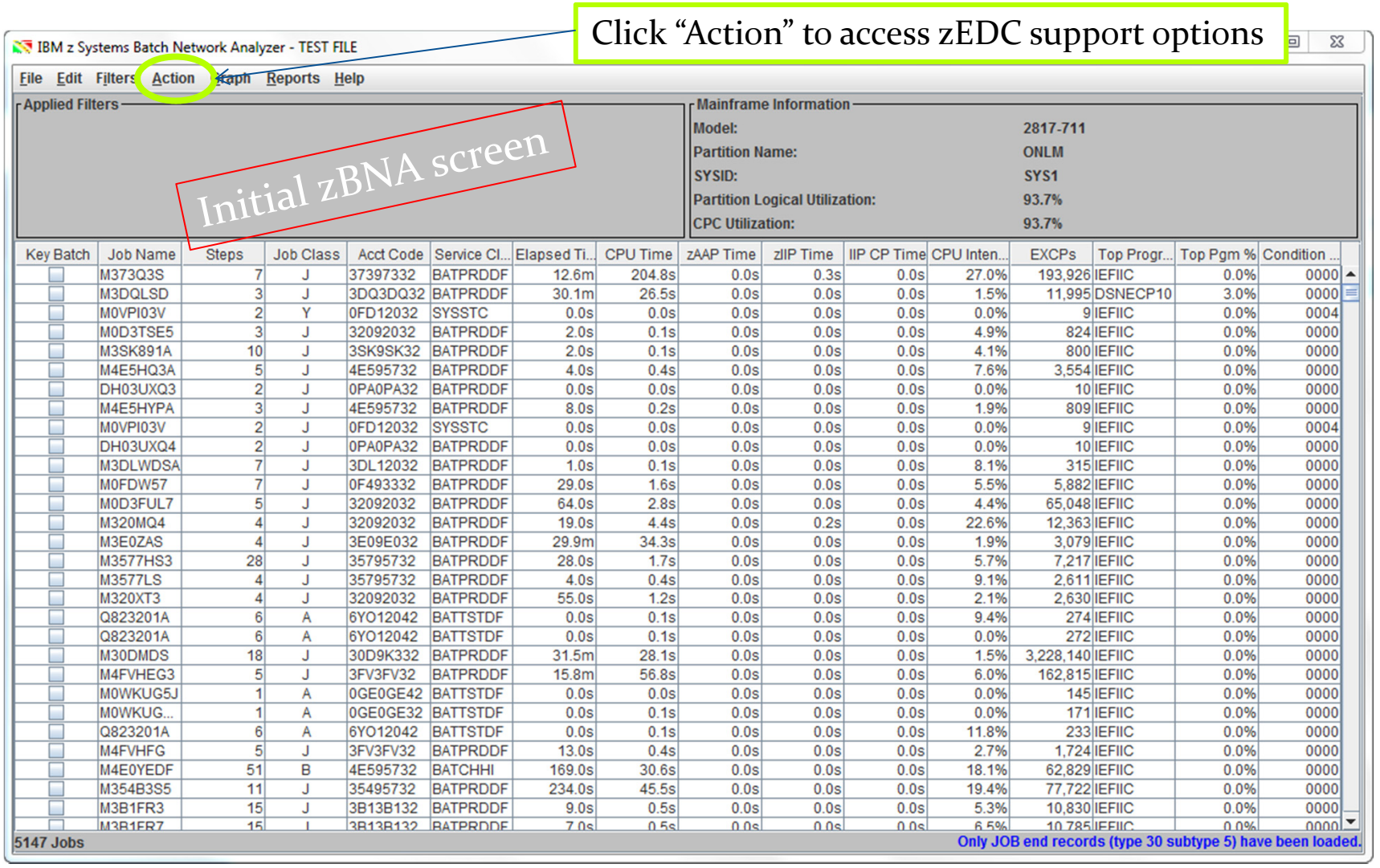
- At this time, the only tool available from IBM to help you estimate the potential cost and savings of zEDC (IBM's zSystems Batch Network Analyzer (zBNA)) only handles sizing for BSAM/QSAM use of zEDC.
- zBNA can be downloaded at no charge(!) from http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS5132

- While there are other easy-to-implement exploiters of zEDC, there are currently no tools to help you identify the value that zEDC will deliver for them.
  - However if zEDC can be cost justified for use with sequential data sets alone, savings through additional exploitation of zEDC are a bonus.

Identifying potential benefit of zEDC

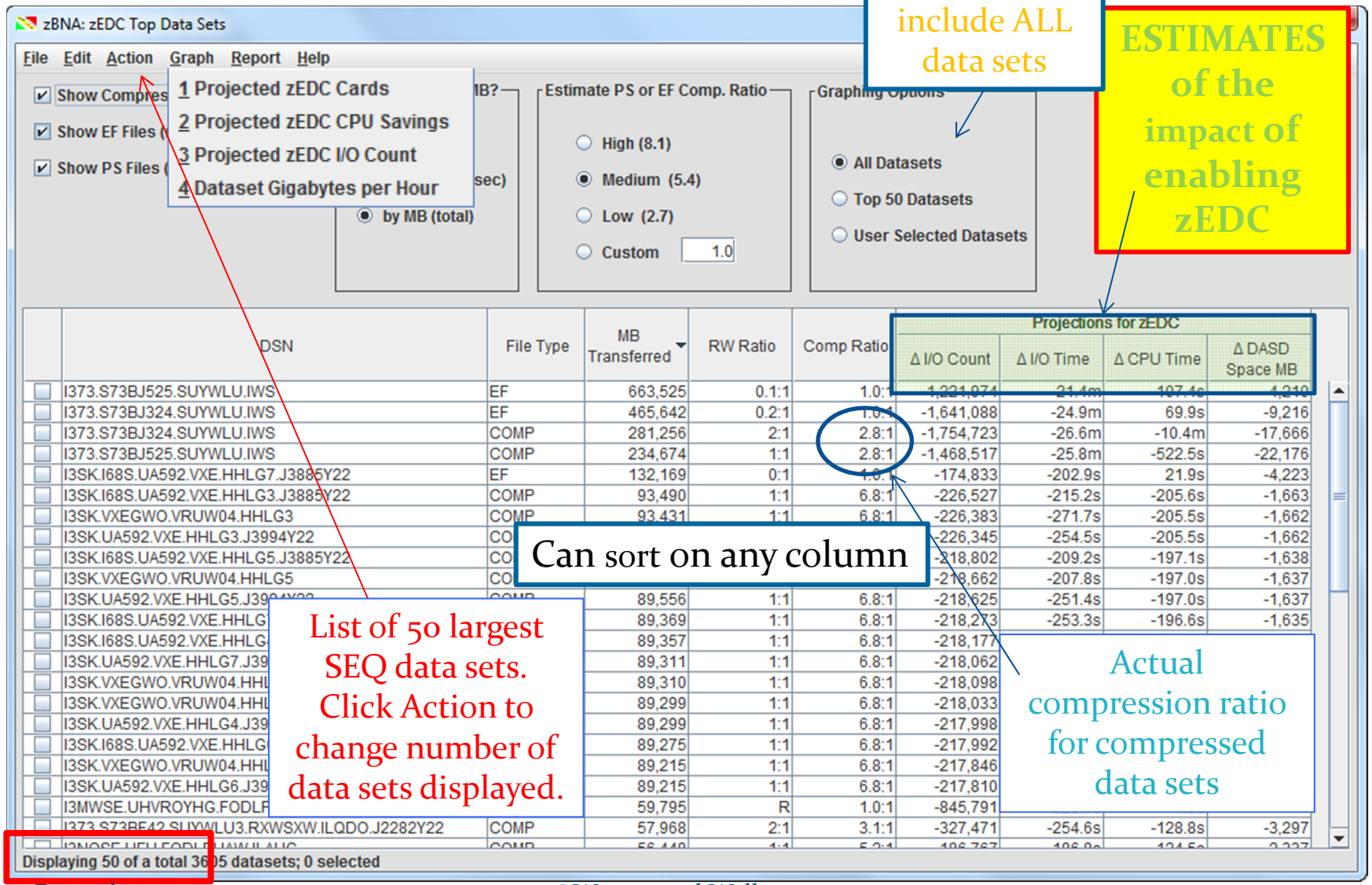
- zBNA uses data from SMF Type 14, 15, 30, 42(6), 70, 72, and 113 records.
  - Run CP3KEXTR against these SMF records to create input file for zBNA. **Input should only contain records from one system for one night** – loading larger volumes of data can result in '878' abends in zBNA.
  - To save time, TERSE the resulting file on z/OS and download that file to your PC. zBNA can input the tersed file directly.
- zBNA has been receiving a stream of enhancements to its zEDC support, so make sure that you download the latest version and review the What's New section on the web site to see the list of recent new functions.
- If you are not familiar with zBNA, download the handouts from http://share.org for John Burg's zBNA hands-on lab – Session **17551** *System z Batch Network Analyzer (zBNA) Tool Hands-on Lab* and get sample data from the IBM Techdocs web site, in the Education section.

Identifying potential benefit of zEDC



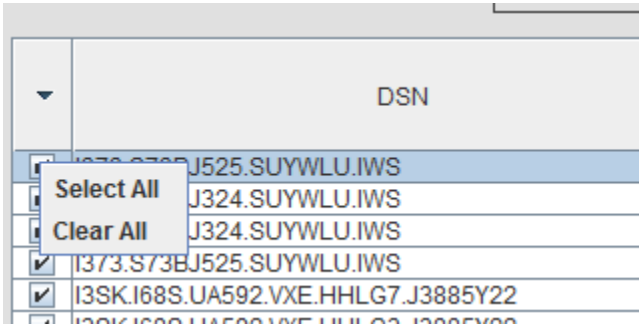
Key Batch	Job Name	Steps	Job Class	Acct Code	Service Cl...	Elapsed Ti...	CPU Time	zAAP Time	zIIP Time	IIP CP Time	CPU Inten...	EXCPs	Top Progr...	Top Pgm %	Condition ...
	M373Q3S	7	J	37397332	BATPRDDF	12.6m	204.8s	0.0s	0.3s	0.0s	27.0%	193,926	IEFIIC	0.0%	0000
	M3DQLSD	3	J	3DQ3DQ32	BATPRDDF	30.1m	26.5s	0.0s	0.0s	0.0s	1.5%	11,995	DSNECP10	3.0%	0000
	M0VPI03V	2	Y	0FD12032	SYSSTC	0.0s	0.0s	0.0s	0.0s	0.0s	0.0%	9	IEFIIC	0.0%	0004
	M0D3TSE5	3	J	32092032	BATPRDDF	2.0s	0.1s	0.0s	0.0s	0.0s	4.9%	824	IEFIIC	0.0%	0000
	M3SK891A	10	J	3SK9SK32	BATPRDDF	2.0s	0.1s	0.0s	0.0s	0.0s	4.1%	800	IEFIIC	0.0%	0000
	M4E5HQ3A	5	J	4E595732	BATPRDDF	4.0s	0.4s	0.0s	0.0s	0.0s	7.6%	3,554	IEFIIC	0.0%	0000
	DH03UXQ3	2	J	0PA0PA32	BATPRDDF	0.0s	0.0s	0.0s	0.0s	0.0s	0.0%	10	IEFIIC	0.0%	0000
	M4E5HYPA	3	J	4E595732	BATPRDDF	8.0s	0.2s	0.0s	0.0s	0.0s	1.9%	809	IEFIIC	0.0%	0000
	M0VPI03V	2	J	0FD12032	SYSSTC	0.0s	0.0s	0.0s	0.0s	0.0s	0.0%	9	IEFIIC	0.0%	0004
	DH03UXQ4	2	J	0PA0PA32	BATPRDDF	0.0s	0.0s	0.0s	0.0s	0.0s	0.0%	10	IEFIIC	0.0%	0000
	M3DLWDSA	7	J	3DL12032	BATPRDDF	1.0s	0.1s	0.0s	0.0s	0.0s	8.1%	315	IEFIIC	0.0%	0000
	M0FDW57	7	J	0F493332	BATPRDDF	29.0s	1.6s	0.0s	0.0s	0.0s	5.5%	5,882	IEFIIC	0.0%	0000
	M0D3FUL7	5	J	32092032	BATPRDDF	64.0s	2.8s	0.0s	0.0s	0.0s	4.4%	65,048	IEFIIC	0.0%	0000
	M320MQ4	4	J	32092032	BATPRDDF	19.0s	4.4s	0.0s	0.2s	0.0s	22.6%	12,363	IEFIIC	0.0%	0000
	M3E0ZAS	4	J	3E09E032	BATPRDDF	29.9m	34.3s	0.0s	0.0s	0.0s	1.9%	3,079	IEFIIC	0.0%	0000
	M3577HS3	28	J	35795732	BATPRDDF	28.0s	1.7s	0.0s	0.0s	0.0s	5.7%	7,217	IEFIIC	0.0%	0000
	M3577LS	4	J	35795732	BATPRDDF	4.0s	0.4s	0.0s	0.0s	0.0s	9.1%	2,611	IEFIIC	0.0%	0000
	M320XT3	4	J	32092032	BATPRDDF	55.0s	1.2s	0.0s	0.0s	0.0s	2.1%	2,630	IEFIIC	0.0%	0000
	Q823201A	6	A	6YO12042	BATTSTDF	0.0s	0.1s	0.0s	0.0s	0.0s	9.4%	274	IEFIIC	0.0%	0000
	Q823201A	6	A	6YO12042	BATTSTDF	0.0s	0.1s	0.0s	0.0s	0.0s	0.0%	272	IEFIIC	0.0%	0000
	M30DMDS	18	J	30D9K332	BATPRDDF	31.5m	28.1s	0.0s	0.0s	0.0s	1.5%	3,228,140	IEFIIC	0.0%	0000
	M4FVHEG3	5	J	3FV3FV32	BATPRDDF	15.8m	56.8s	0.0s	0.0s	0.0s	6.0%	162,815	IEFIIC	0.0%	0000
	M0WKUG5J	1	A	0GE0GE42	BATTSTDF	0.0s	0.0s	0.0s	0.0s	0.0s	0.0%	145	IEFIIC	0.0%	0000
	M0WKUG...	1	A	0GE0GE32	BATTSTDF	0.0s	0.1s	0.0s	0.0s	0.0s	0.0%	171	IEFIIC	0.0%	0000
	Q823201A	6	A	6YO12042	BATTSTDF	0.0s	0.1s	0.0s	0.0s	0.0s	11.8%	233	IEFIIC	0.0%	0000
	M4FVHFG	5	J	3FV3FV32	BATPRDDF	13.0s	0.4s	0.0s	0.0s	0.0s	2.7%	1,724	IEFIIC	0.0%	0000
	M4E0YEDF	51	B	4E595732	BATCHHI	169.0s	30.6s	0.0s	0.0s	0.0s	18.1%	62,829	IEFIIC	0.0%	0000
	M354B3S5	11	J	35495732	BATPRDDF	234.0s	45.5s	0.0s	0.0s	0.0s	19.4%	77,722	IEFIIC	0.0%	0000
	M3B1FR3	15	J	3B13B132	BATPRDDF	9.0s	0.5s	0.0s	0.0s	0.0s	5.3%	10,830	IEFIIC	0.0%	0000
	M3B1FR7	15	J	3B13B132	BATPRDDF	7.0s	0.5s	0.0s	0.0s	0.0s	6.5%	10,785	IEFIIC	0.0%	0000

Identifying potential benefit of zEDC

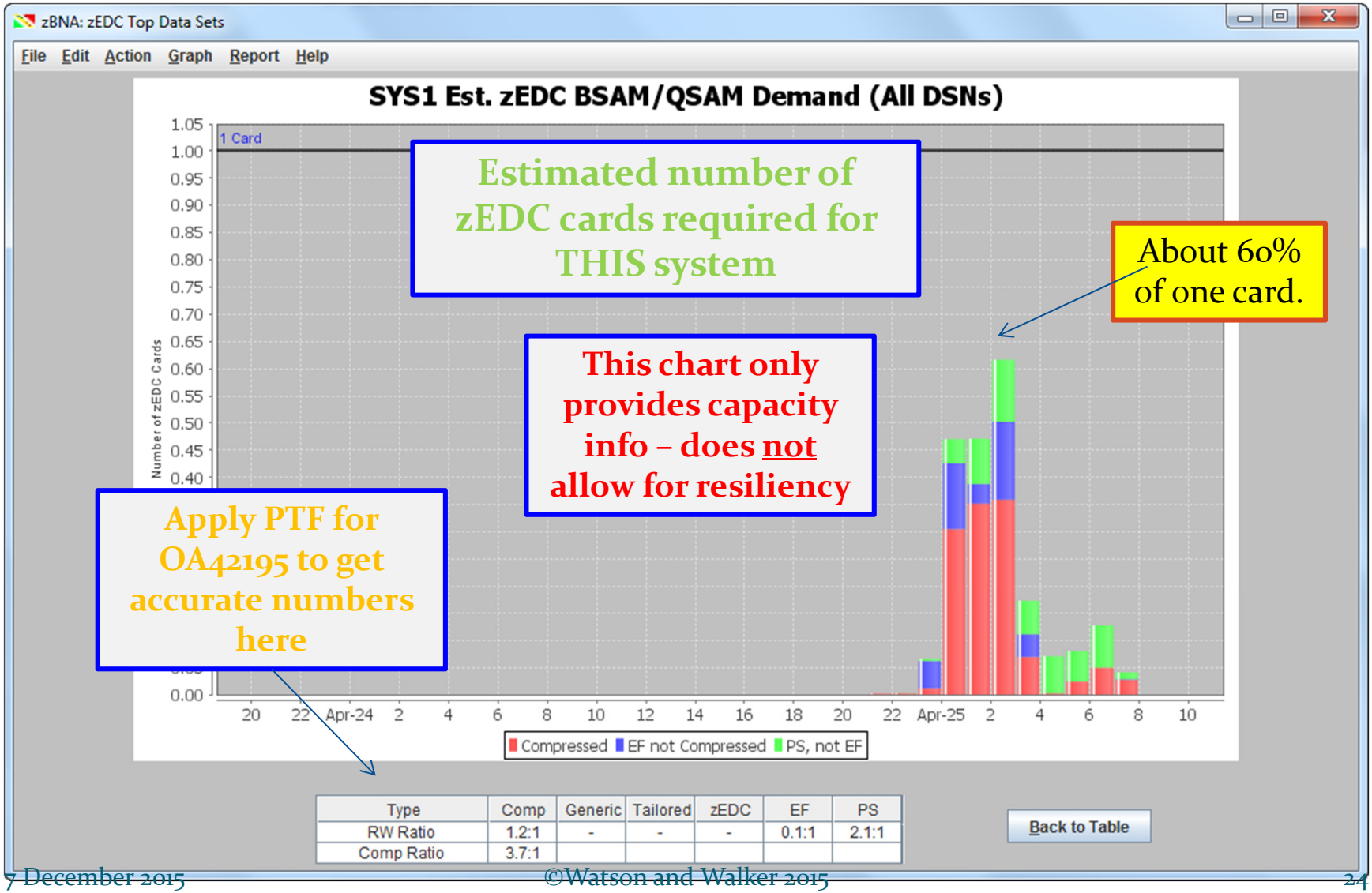


Identifying potential benefit of zEDC

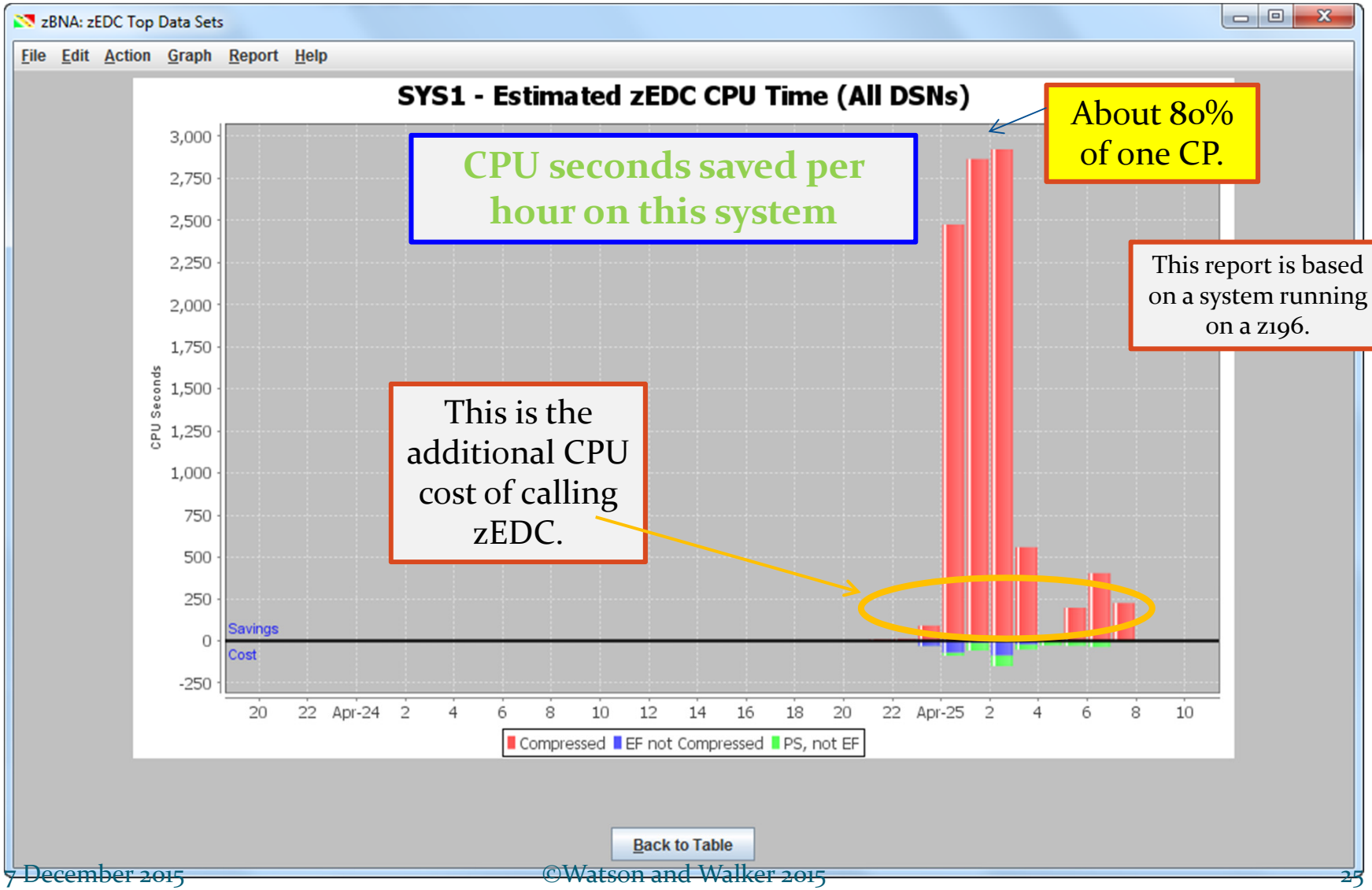
- Be aware that SAS does not support zEDC compression for its databases, however the SAS PDBs look like ordinary sequential data sets to zBNA, so be careful to exclude those from the calculations.
- Similarly, any other 'sequential' data set that is processed by an access method other than BSAM or QSAM will not support zEDC, so be aware of those in the zBNA results.
- To select just a subset of data sets, put your mouse over any of the check boxes and click right mouse button. Select 'select all', and then deselect any that you want to exclude.



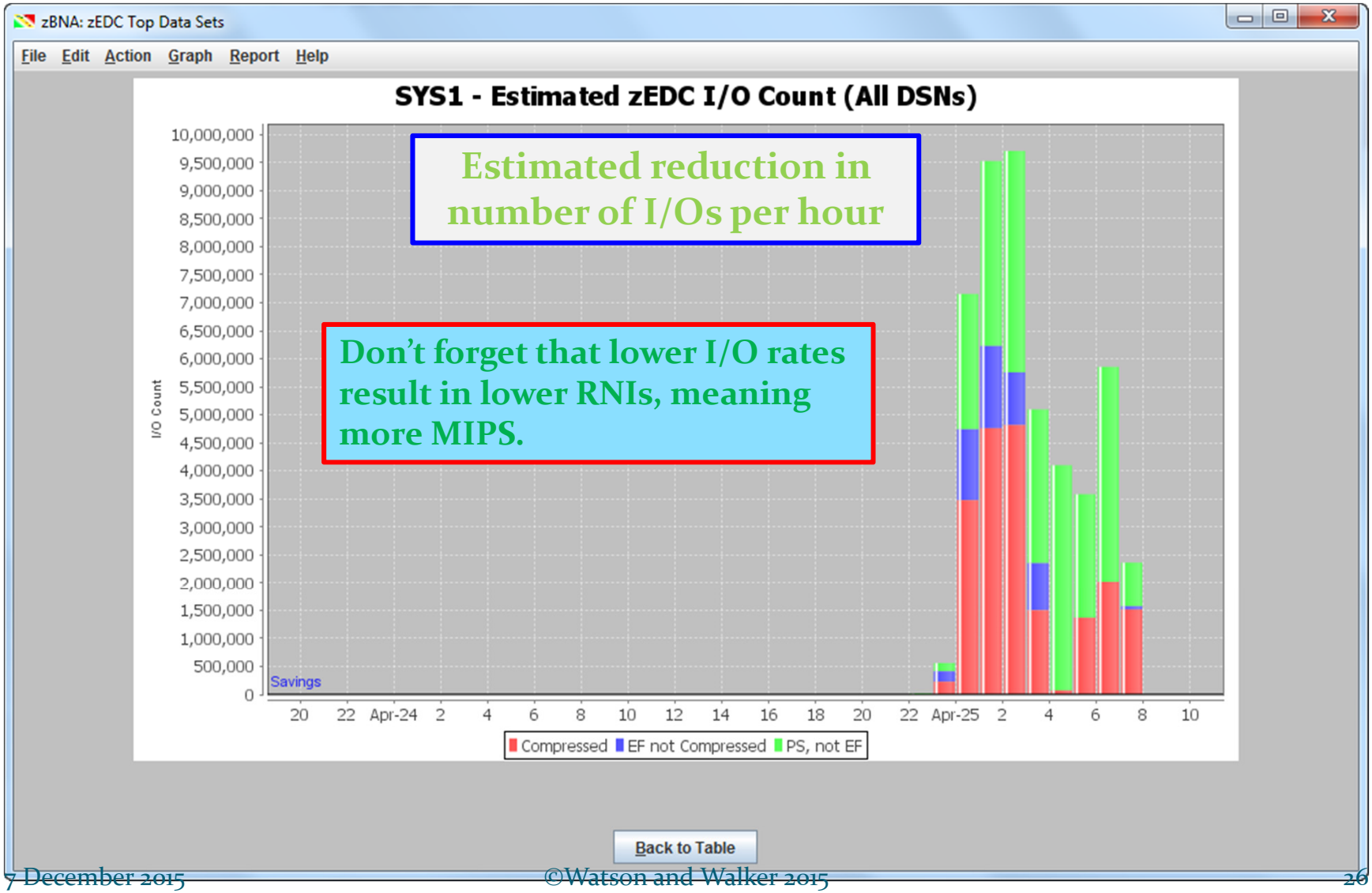
Identifying potential benefit of zEDC



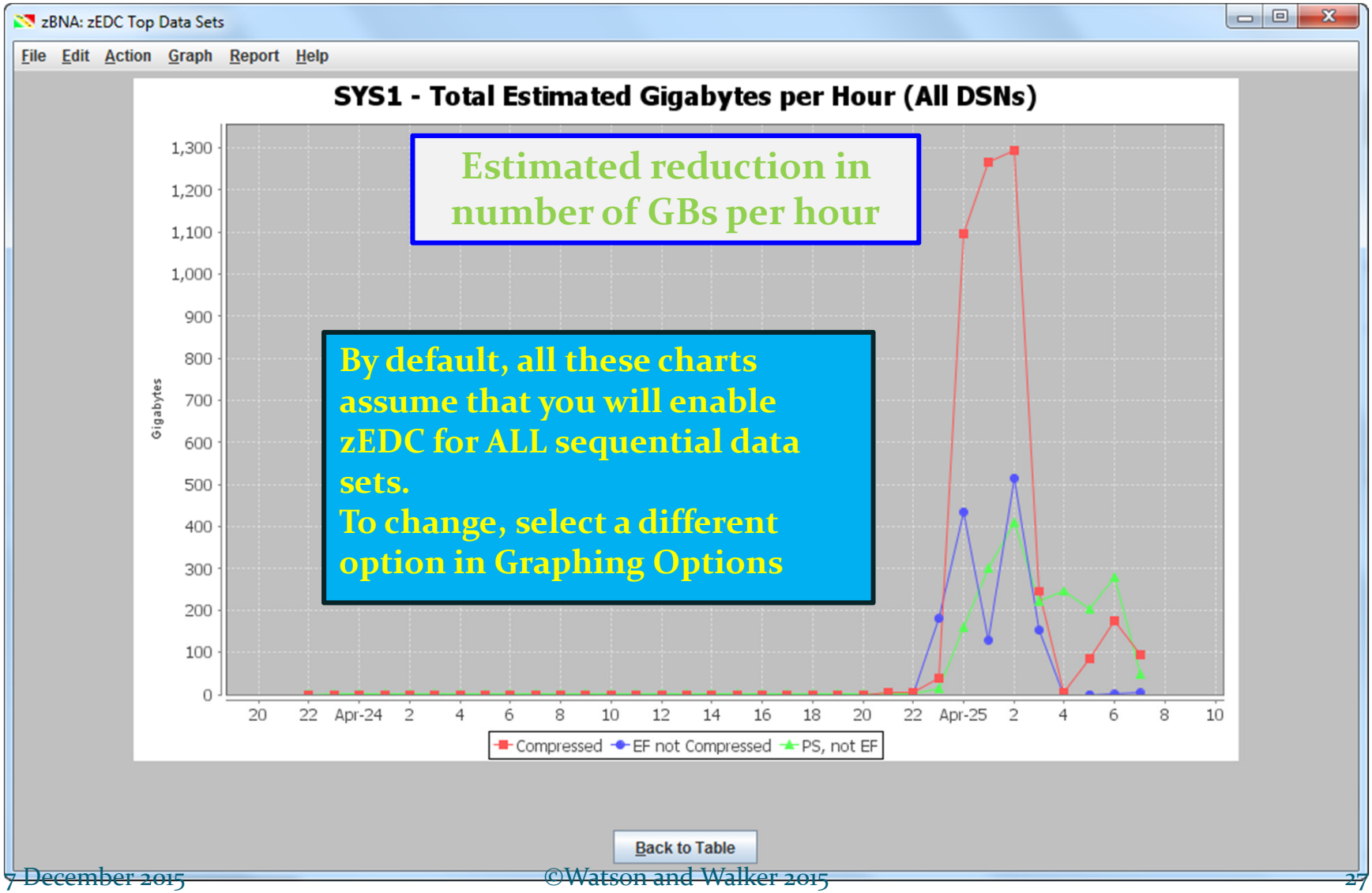
Identifying potential benefit of zEDC



Identifying potential benefit of zEDC



Identifying potential benefit of zEDC



Identifying potential benefit of zEDC

- Few tips for using zBNA:
  - zEDC provides 3 potential savings:
    - DASD and tape space savings – should apply to EVERYONE
    - Free up CP capacity – should apply to EVERYONE *that uses compression today*.
    - Reduced software bills.
      - To achieve this benefit, you must be using compression today, *and* your peak Rolling 4-Hour Average must coincide with your peak compression activity.
      - But don't get hung up on just this savings – the other two savings are real as well.
  - Remember that zBNA is only looking at one exploiter of zEDC. It does not do anything for SMF or zlib or MQ or Connect Direct or …. So if you plan to use additional exploiters, the benefit of zEDC is likely to be larger.

Configuring zEDC using HCD

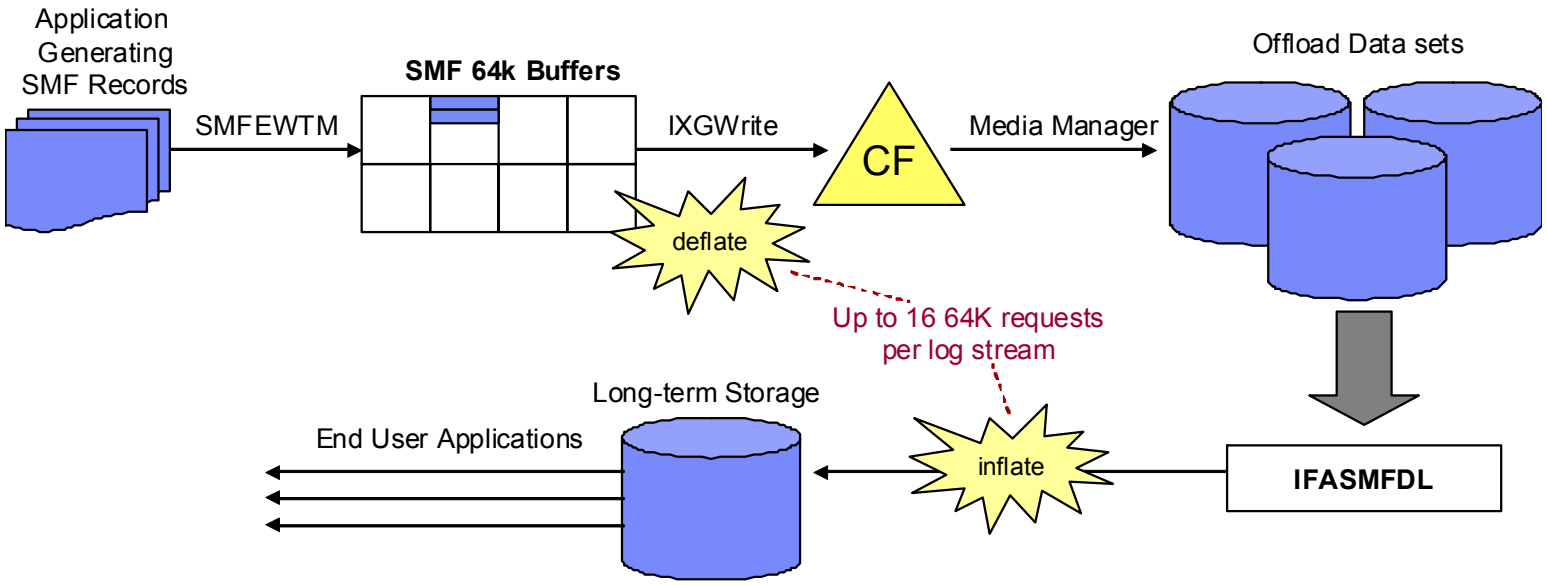
- The ITSO have created 2 excellent videos that lead you step-by-step through the process of defining a zEDC card in HCD. For info on how to add a zEDC card (in less than 7 minutes!), refer to:
  - https://www.youtube.com/watch?v=5We571gvh5o and
  - https://www.youtube.com/watch?v=wZbtM77ubrs
  - Recommend using PFIDs starting with an even digit for one Resource Group, and an odd digit for the other Resource Group. Then make sure that every LPAR is connected to at least one card in each Resource Group (recommended minimum of 2 cards).
- After the hardware setup is complete, you need to update the IFAPRDxx member to enable zEDC (this requires an IPL).
- IF you have unauthorized programs using the zlib interface to zEDC, they must have READ access to the zEDC SAF profile (FPZ.ACCELERATOR.COMPRESSION.)

How to enable zEDC exploitation – SMF Log streams

- The first exploiter of zEDC was SMF (when in log stream mode). This is a very popular starting point because the implementation effort is trivial and the savings can be significant.
  - SMF is also nice because the heaviest volume of SMF records is typically during the online day when other use of zEDC (sequential data sets, HSM, DSS) is typically light.

- To exploit zEDC, SMF must be writing its records to log streams. SYS1.MAN data sets are not and will not be supported for zEDC compression. If you are already in log stream mode, you are 99.99% of the way there.
- To enable compression of an SMF log stream, add the COMPRESS keyword to the LSNAME or DEFAULTLSNAME definition in SMFPRMxx:
  - DEFAULTLSNAME(IFASMF.DEFAULT,**COMPRESS**) DSPSIZMAX(200M)
  - You can have a mix of some log streams that *are* using zEDC and ones that are *not*.

How to enable zEDC exploitation – SMF Log streams



- Each log block has an indication of whether it contains zEDC-compressed data or not. When IFASMFDL reads the log block, it knows whether it needs to be decompressed or not. This allows you to have a mix of compressed and not-compressed log blocks in the same log stream (allows for a phased implementation of zEDC compression of shared SMF log streams across multiple systems).

How to enable zEDC exploitation – SMF Log streams

- Summary of exploiting zEDC for SMF data
  - That's it – just one new keyword in your SMFPRMxx member, no additional changes required.
  - Compression of a log stream can be turned on and off dynamically.
    - But be aware that if you want to turn it OFF, *removing* the COMPRESS keyword will not achieve that – you must change COMPRESS to NOCOMPRESS.
  - IFASMFDL is the only way to read the SMF log stream, and it automatically detects whether a log block is compressed or not, meaning that no JCL or Parm changes are required.
    - See http://www.ibm.com/support/docview.wss?uid=isg3T1022540 for info about IFASEXIT and zEDC – APAR is expected to add this support in 1Q16.
  - To get the best value from zEDC, your SMF offload sequential data sets should use a data class that specifies the use of zEDC compression.

How to enable zEDC exploitation – BSAM/QSAM

- Traditional DFSMS compression for sequential data sets is enabled by assigning the correct SMS data class to a data set. One of the attributes in the data class definition is whether the data set should be compressed, and if so, what algorithm should be used:

```
  Panel  Utilities  Scroll  Help
------------------------------------------------------------------------------
                              DATA CLASS DISPLAY                 Page 2 of 5
Command ===> _

CDS Name  . . . . . : SYS1.WTSCPLX8.SCDS
Data Class Name . . : SAMGEN

Data Set Name Type  . . . . . : EXTENDED
  If Extended . . . . . : REQUIRED
  Extended Addressability . . : NO
  Record Access Bias  . . . . : USER
  RMODE31 . . . . . :
Space Constraint Relief . . . : NO
  Reduce Space Up To (%)  . . :
  Guaranteed Space Reduction  : NO
  Dynamic Volume Count  . . . :

Compaction  . . . . . : GEN
Spanned / Nonspanned  . . . . :


Use UP/DOWN Command to View other Panels;
Use HELP Command for Help; Use END Command to Exit.
```

How to enable zEDC exploitation – BSAM/QSAM

- To enable zEDC compression, we recommend setting up a new data class for testing purposes. That data class should be defined with zEDC REQUIRED:

```
HELP----------------------------  COMPACTION  ----------------------------HELP
COMMAND ===> _
                                                                More:    - +
        ZR      The system will fail the allocation request if the zEDC
                function is not supported by the system or the minimum
                allocation amount requirement is not met.
        ZP      The system will not fail the allocation request but rather
                create either a tailored compressed data set if the zEDC
                function is not supported by the system or create a
                non-compressed extended format data set if the minimum
                allocation amount requirement is not met.
        blank   Data sets are not compressed. Tape volumes may
                be compacted depending on what was specified by the user
                on JCL/dynamic allocation, the installation with the COMPACT
                option in parmlib member DEVSUPxx, or the allocated hardware
                model. This is the default.

    For example:

        Compaction . . . . . . . N       (Y, N, T, G, ZR, ZP or blank)

    Use ENTER to continue, END to exit help.
```

How to enable zEDC exploitation – BSAM/QSAM

- You then conduct your testing by specifying the correct data class when you allocate the data set.
- Testing should include allocation, deletion, extending the data set, filling the data set, partial release, single strip, multiple stripe, varying buffer numbers, backup, migration, recall, restore…..
- When testing is complete, update the previous compression data classes to indicate ZR (zEDC required), and (if appropriate) update the COMPRESS parameter in IGDSMSxx.

zEDC implementation tips

- If the system has the required releases and PTFs to support zEDC use by BSAM/QSAM, that meets the criteria of zEDC Required EVEN IF THE LPAR DOES NOT HAVE ACCESS TO A zEDC CARD. So the job ends with RC=0, with NO joblog message to tell you that zEDC was not available, and with a data set that is in zEDC-format, but whose contents are **not** compressed.
  - Save yourself hours of frustration by issuing D PCIE command to be sure.
- Remember that compressed seq data sets (generic, tailored, or zEDC) must be Extended Format. If you specify DSNTYPE=LARGE, that overrides DSNTYPE=EXT in the data class, meaning that the data set will not be compressed, even if the data class specifies that compression should be used.
- Some zEDC exploiters provide the ability to decompress zEDC-compressed data if zEDC is not available. This should only be considered for use in an emergency as it is EXTREMELY CPU-intensive.

zEDC implementation tips

- **Test software decompression of a compressed data set in an LPAR that doesn't have access to zEDC card. This will convince you of why you want access to zEDC in every LPAR that might touch zEDC-compressed data.**

- Decompressing 5425 track data set WITH zEDC:
  - Elapsed 22.92 Secs
  - TCB 1.88 Secs
  - SRB0.24 Secs

- Decompressing 5425 track data set WITHOUT zEDC:
  - Elapsed 84 Secs
  - TCB 70 Secs (!)
  - SRB.21 Secs

zEDC implementation tips

- What is the impact of zEDC on your chargeback algorithms?
  - YOUR bills may go UP, because you must pay for the hardware and software.
  - Your USERS bills go DOWN, because now they are using less CPU time….
- To address this scenario, IBM have added a new zEDC Usage Statistics section in the SMF Type 30 records. You will now be able to see how much use a jobstep made of zEDC and can adjust its bill accordingly.
  - But remember that you do NOT pay for the "CPU Time" on the zEDC card, so the user's bill should reflect the data center's reduced costs.
- See APARs OA45767 and OA48268 (OPEN)

zEDC implementation tips

- An important zEDC PTF is UA77619 (ABEND11E-0702 OR ABEND002-F6 in job using zEDC). The PTF is available, but one of the pre-reqs is PE.
- However you can force on the pre-req PTF if you specify FREEMAINEDFRAMES(NO) in DIAGxx. For more information, see APAR OA46291.

zEDC implementation tips

- If you are using CICS SMF record compression and/or DB2 record compression today, those functions use software compression, not zEDC.
- Recommend that you turn off CICS and DB2 SMF record compression after you implement zEDC compression for the associated SMF log streams.
  - zEDC will provide at least as good compression, probably more.
  - AND you save the CPU cost in CICS and DB2.

zEDC implementation tips

- When maintenance is applied to zEDC cards (a hardware function), *all* the cards in one of the two Resource Groups will be taken offline.
- To ensure that zEDC remains available, every LPAR should be connected to at least 1 zEDC card *in each Resource Group*.
  - **This means that you should ALWAYS have a minimum of two zEDC cards on a CPC. If you only have one, you may need to pause processing for any workload using zEDC while maintenance is being applied.**
- To cater for the possibility of one of the surviving cards failing while maintenance is being applied to the other Resource Group, IBM recommend that each LPAR should be connected to 4 zEDC cards.
- Apply the PTF for APAR OA48434 – this is a Health Check to warn you if the LPAR doesn't have access to more than one zEDC card. Enable the check and add the check message to your automation to ensure that the appropriate personnel are notified if a card becomes unavailable.

Monitoring zEDC performance

- The bandwidth of the zEDC card is over 1GB/sec.
  - Compare this to about 300 MB/sec for the CMPSC instruction and between 50 and 100MB/sec for zlib.
- In our test with 16 parallel compression jobs running, we couldn't drive the utilization of the card above 34%. So, for most customers, 2 cards should provide sufficient capacity.
- If you find that utilization of the card is increasing, that is a GOOD THING, because it means that work that would previously have run on a general purpose CP (and that would have been counted in the determination of your software bills) is now being processed on a cheaper, faster processor AND is not impacting your software bills.

Other monitoring information for zEDC

- How do you know when you need more zEDC capacity?
  - The zEDC card works on a simple round robin polling mechanism, checking each LPAR it is connected to, to see if they have any work for it.
    - If no work, it tries the next LPAR.
    - As the card utilization increases, the interval between when it polls each LPAR gets larger, potentially resulting in queueing for longer in the z/OS images.
      - The queue time is reported in the RMF PCIE report ("Request Queue Time").
      - Unlike zIIPs, there is no overflow to general purpose CPs – if the card is overloaded, users of its services will simply observe longer queue times.
  - So, the point at which you need more zEDC capacity is determined by how much queue time you are willing to tolerate.
    - Queue time is determined by normal queue time formula. With more 'servers' (more zEDC cards), you can run at higher utilizations before queue time increases.
  - zEDC is NOT like zIIP. If the card is full, compression will NOT overflow back to the general purpose CPs.

zEDC prerequisites

- zEDC cards are available on zEC12, zBC12, z13, and later.
  - Up to 8 cards per CPC, 2 per PCIe drawer.
  - Max of 15 LPARs per card.
- Exploitation requires z/OS 2.1 or later.
  - Ability to read zEDC-compressed files using software decompression is available on z/OS 1.13 and later.
  - Requires support (appropriate releases or PTFs) in the exploiters that you want to use as well.
  - Make sure that you monitor the IBM.FUNCTION.zEDC FIXCAT for required PTFs.
- Using software to decompress data that was compressed using zEDC is EXTREMELY CPU-intensive and should only be used in exceptional situations.
- Data sets compressed using BSAM/QSAM zEDC support must be SMS managed.
  - Data class should specify zEDC Preferred or zEDC Required
  - Data sets compressed with traditional DFSMS compression also must be SMS-managed.

Summary

- Initial take-up of zEDC was slow because you must have zEC12 or later and z/OS 2.1 or later AND you want those on every system that will share the data, and it is only now that those configurations are becoming common.
- Customer experiences so far have been very positive. One customer described zEDC as 'a game changer'.
- Download and run zBNA – it has many uses in addition to planning for zEDC, so everyone should have it anyway.
- Once the hardware and software are in place, the implementation of zEDC is simple.
- Recommend to start with SMF, then move on to DFSMSdss and hsm, and then all large sequential data sets.
- Tell anyone that is responsible for large file transfers to and from z/OS about the benefits they might be able to get from zEDC.

zEDC Reference Information

- IBM Redbook SG24-8259, *Reduce Storage Occupancy and Increase Operations Efficiency with IBM zEnterprise Data Compression*
- IBM RedPaper, REDP-5158, *zEDC Compression: DFSMShsm Sample Implementation*
- z/OSMF Workflow IBM: z/OS V2R1 zEnterprise® Data Compression Setup Workflow
- zEDC Product Manual SA23-1377, *z/OS MVS Programming: Callable Services for High-Level Languages*
- IBM Manual SA23-1380, *z/OS MVS Initialization and Tuning Reference*
- IBM Hot Topics August 2014, *Save BIG with QSAM/BSAM compression by using zEDC* and *All aboard with zEDC.*
- IBM Hot Topics August 2013, *z Enterprise Data Compression Express*
- IBM Journal of R&D, Volume 59, Number 4/5, July/Sep. 2015, *Integrated high-performance data compression in the IBM z13*
- *For a list of zEDC articles created by its lead designer, refer to https://www.linkedin.com/pub/anthony-sofia/4/6aa/713*
- Multiple excellent SHARE presentations by Anthony Sofia, Barbara McDonald, Cecilia Lewis, and Glenn Wilcock.

Thank you!

- Thank you for your time and questions. And if you are going to be at SHARE in San Antonio, drop by to say hello to Cheryl and me.