Rocket
LEGACY POWERS LEGENDARY

Debugging CICS Storage Violations Using IPCS

Ezriel Gross, Rocket Software

egross@rocketsoftware.com

Tuesday, November 16, 2021

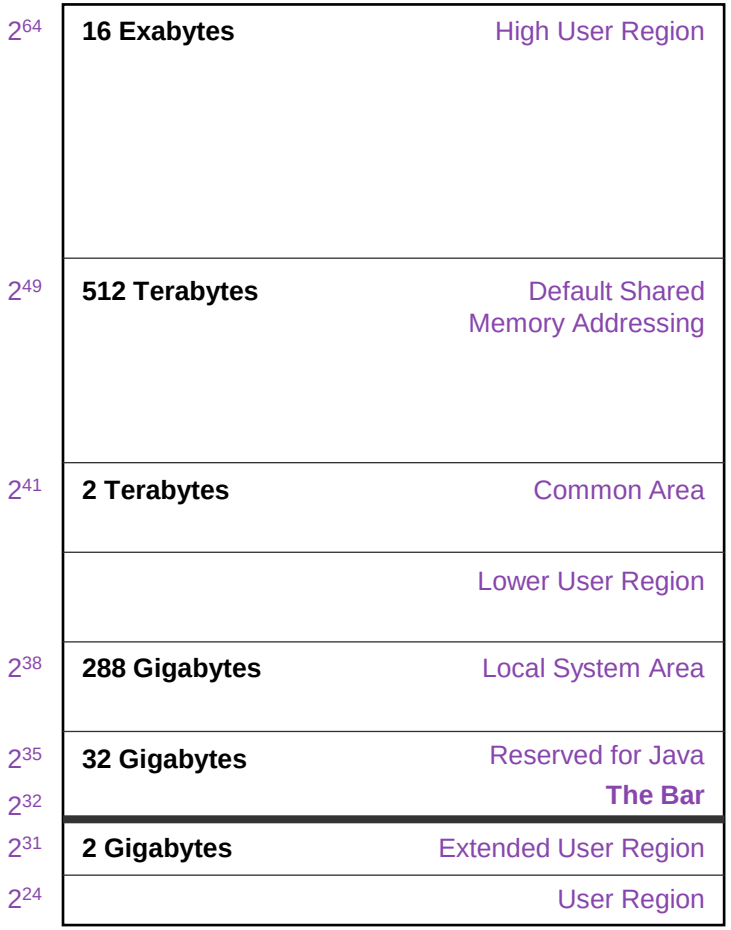
11:30 AM-12:30 PM EST

Agenda

1. CICS Storage Management
2. What is a storage violation?
3. Causes of storage violations
4. Protecting Storage in CICS
   - CICS Provided Facilities
5. Storage Manager internals
6. Storage violation dump analysis
   - Useful domains for debugging Storage violations
   - IPCS Commands to view relevant domain summaries and data

CICS Storage Management

CICS Storage management: Address Space



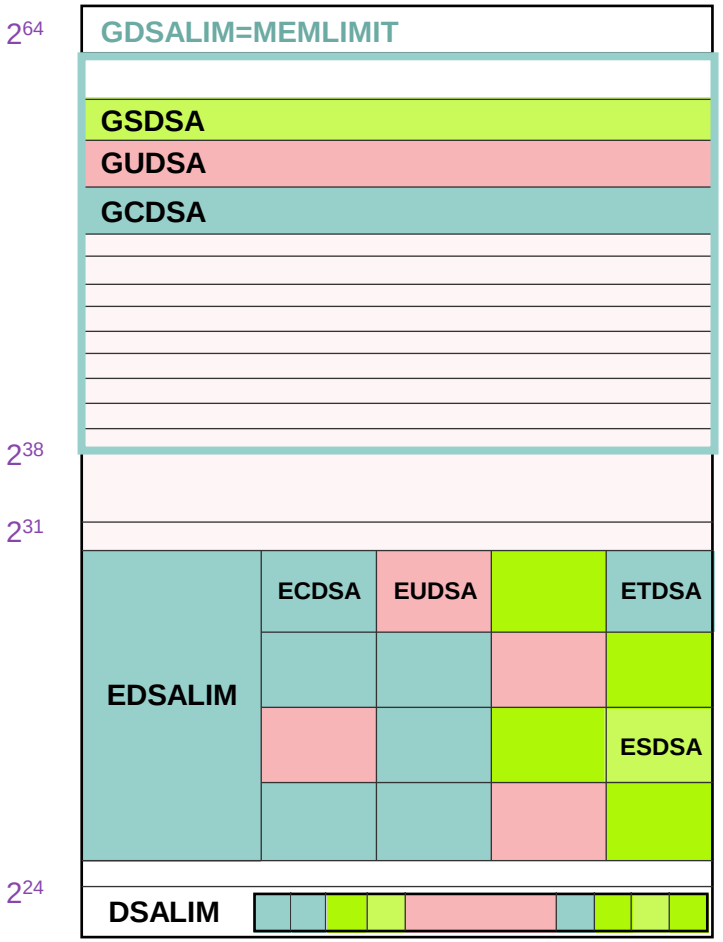
CICS V5

"A 64-bit address space is 8 billion times the size of a 2-gigabyte address space"

MEMLIMIT: Limited the amount of usable virtual storage available to an address space in the above the bar low & high user regions

REGION: Limits the amount of virtual storage available in the below the bar user region & extended user region

CICS Storage management: Dynamic Storage



CICS Initialization sets GDSALIM to MVS **MEMLIMIT** value

If **MEMLIMIT** is below 10 Gb CICS Abend with DFHSM0602

CICS Initialization reserves 31-bit virtual storage based on EDSALIM

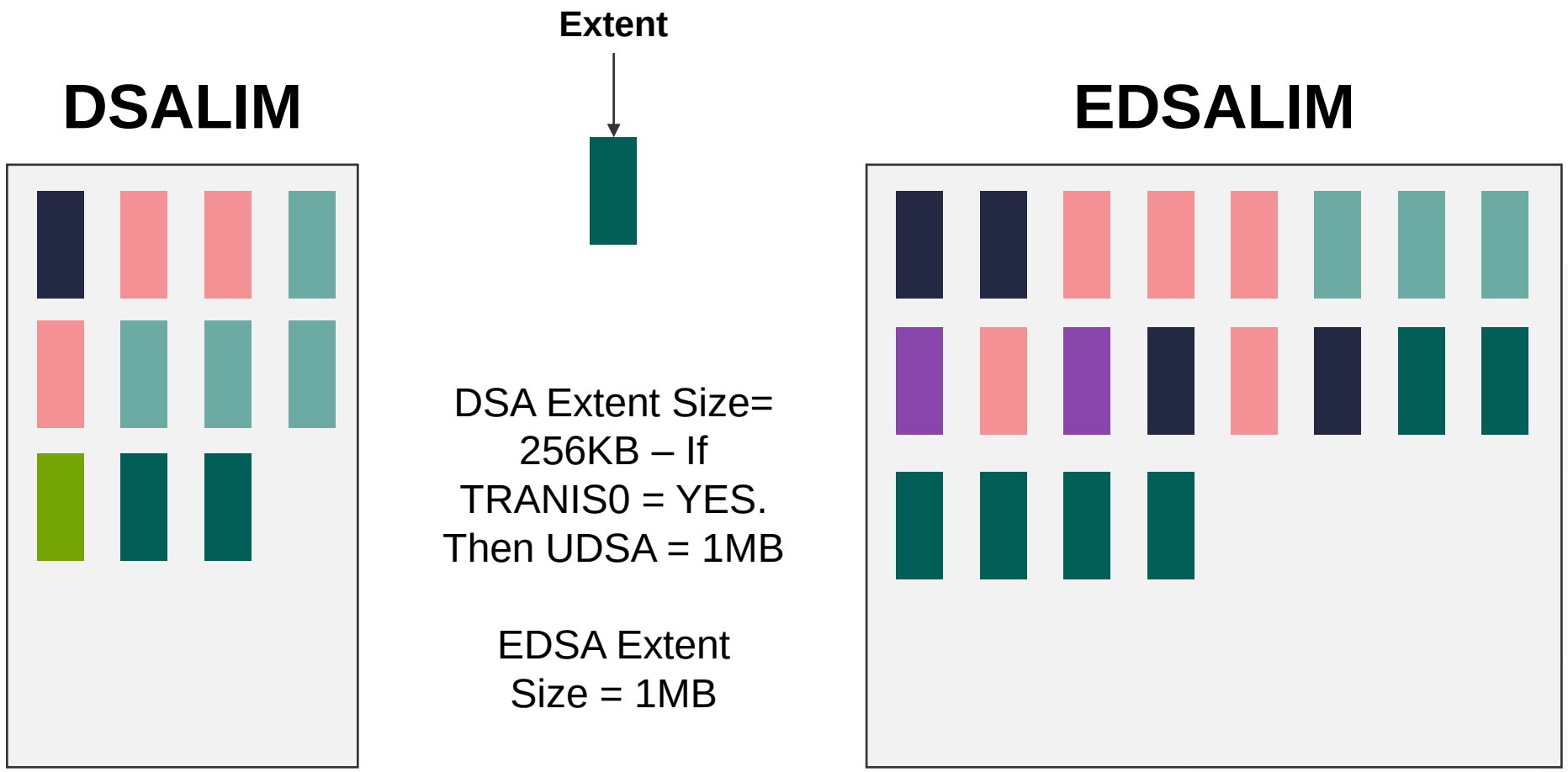
CICS Initialization reserves 24-bit virtual storage based on DSALIM

CICS Storage management: DSAs

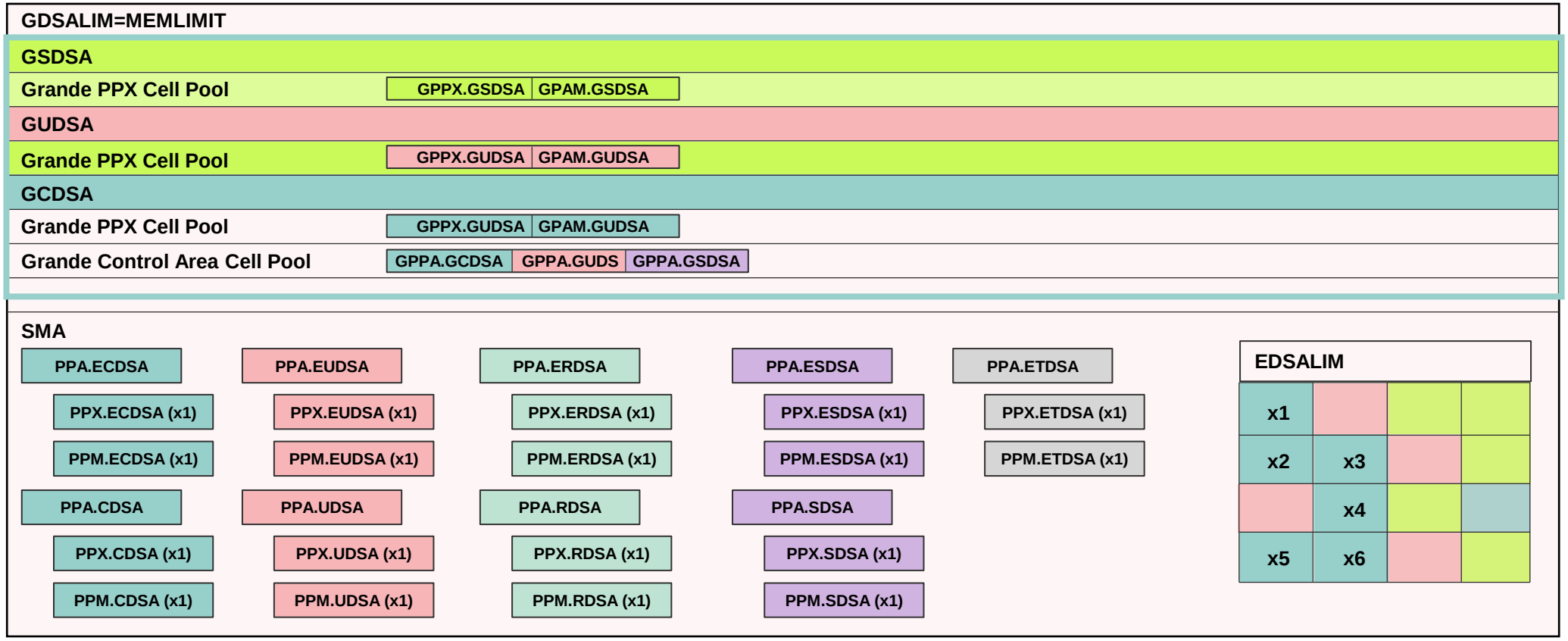
```
I DSA
STATUS:
    Sosabovebar(Notsos)          Memlimit(Nolimit)
    Sosaboveline(Notsos)          Gcdsasize(1G)
    Sosbelowline(Notsos)          Gsdsasize(0)
                                  Gudsasize(0)

  Dsalimit( 05242880 )
   Cdsasize(00262144)
   Rdsasize(00262144)
   Sdsasize(00262144)
   Udsasize(01048576)

  Edsalimit( 1048576000 )
   Ecdsasize(0016777216)
   Erdsasize(0039845888)
   Esdsasize(0001048576)
   Etdsasize(0001048576)
   Eudsasize(0002097152)
```

```
SYSID=FWAR APPLID=FUWFWAR
```

CICS Storage management: DSA Extents

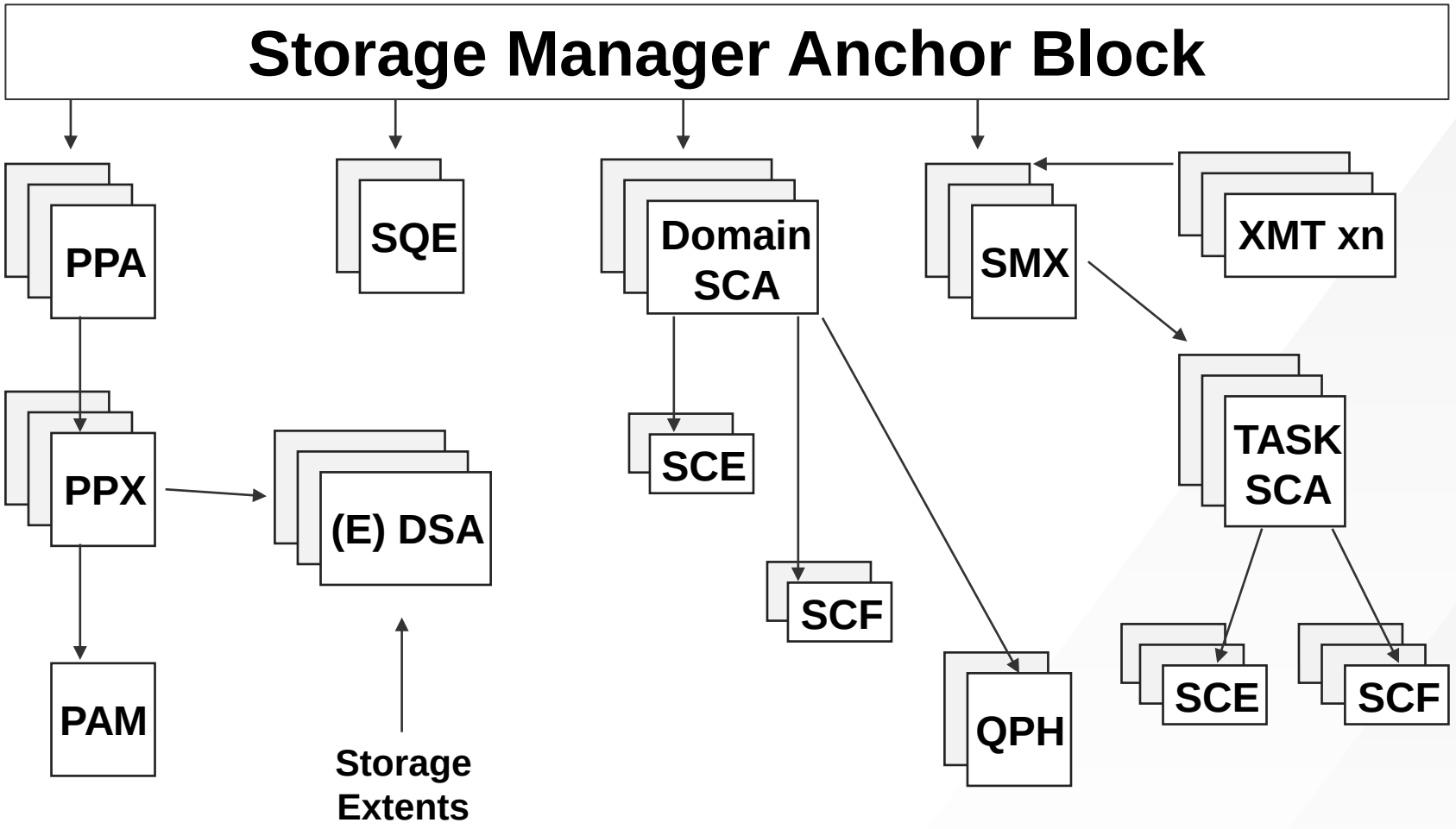


CICS Storage management: DSA Control Blocks



TS20626

CICS Storage management: Control Blocks



SCE/SCF- Used for Variable Storage
QPH- Used for Fixed Length Storage

Domain subpools versus task subpools

Domain subpools

Domain subpools tend to get allocated early on in CICS initialization and remain allocated for the life of CICS. The domain subpool ids are not fixed and can change on any run of CICS.

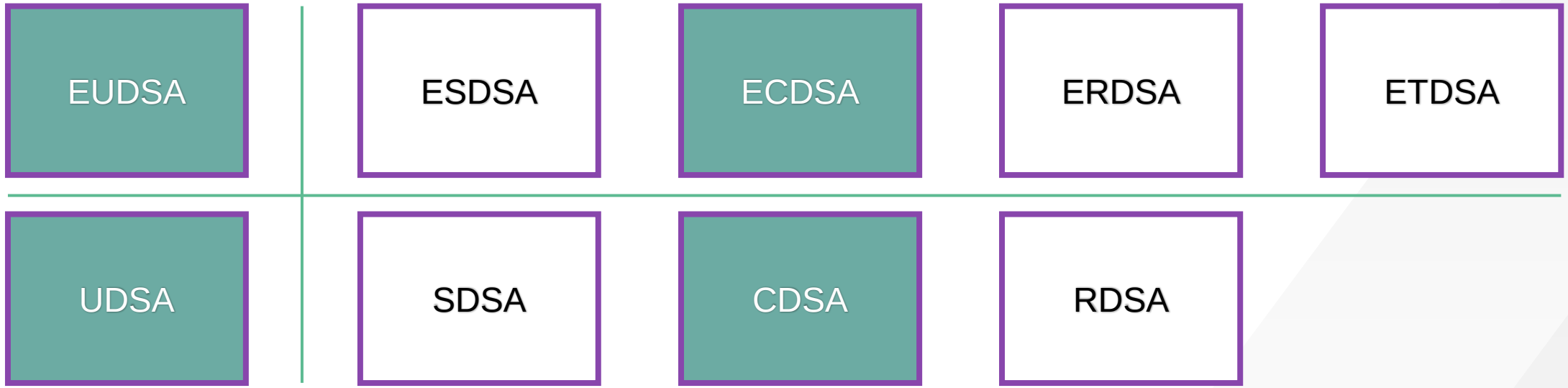
Life of task storage

- Task subpools are dynamically created and deleted for each task in the system as required.
- The task subpool ids are fixed for any run of CICS.

Let’s have a look at a dump and see

What is a storage violation?

CICS DSAs used for life of task storage



- CICS manages storage in DSAs above the Bar (not shown), above the line and below the line
- Only the User and CICS DSAs can contain task lifetime storage with storage violation detection facilities
- **TASKDATAKey = User | CICS** **TASKDATALoc = Below | Any**

Task subpools

SMX Addr	Tran #	Tran Tkn	Name	Id	Loc	Acc	Gets	Frees
5F506064	0000003	20C07400	M0000003	0001	B	C	0	0
			C0000003	0003	A	C	0	0
			G0000003	0005	P	C	0	0
			B0000003	0002	B	U	0	0
			U0000003	0004	A	U	0	0
			H0000003	0006	P	U	0	0
5F5060A8	0000005	20C07A00	M0000005	0001	B	C	0	0
			C0000005	0003	A	C	0	0
			G0000005	0005	P	C	0	0
			B0000005	0002	B	U	0	0
			U0000005	0004	A	U	0	0
			H0000005	0006	P	U	0	0

CICS will allocate 6 task subpools for every task in the system

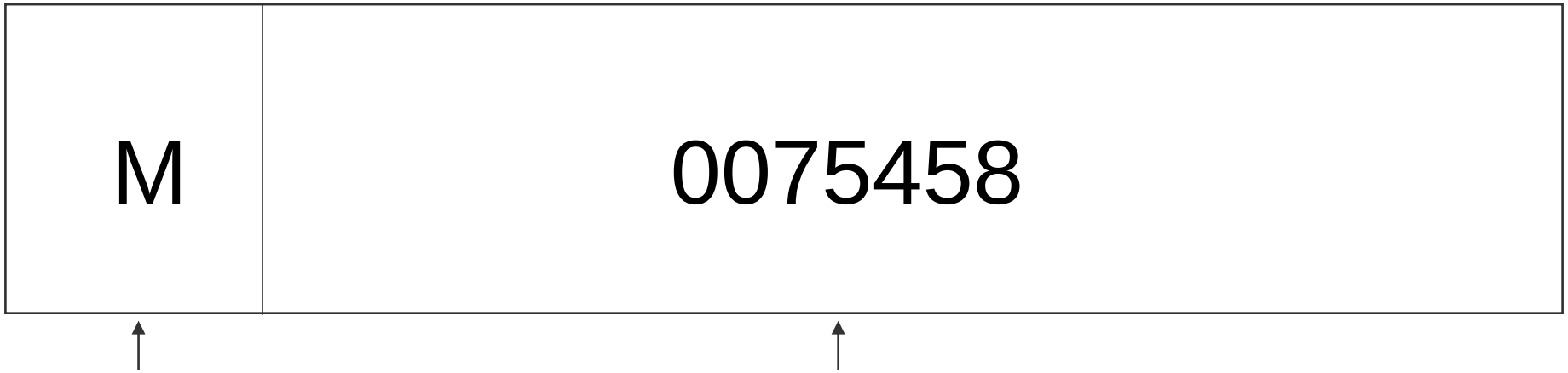
Each subpool is for either the User or CICS DSA in all locations (Above / Below)

These subpools are exclusive to the task for transaction storage requests

The subpool names are 8 characters long, a letter followed by the task number

The subpool name is reused for the Storage Check Zone (SCZ)

Storage Check Zones - SCZ



M = CICS Macro Below
C = CICS Above
B = User Below
U = User Above
G = CICS Above the Bar
H = USER Above the bar

Storage Check Zones - SCZ

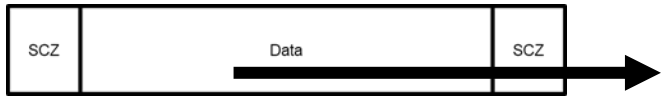
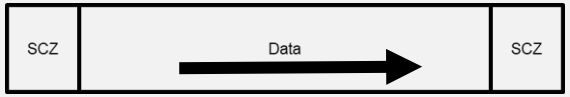
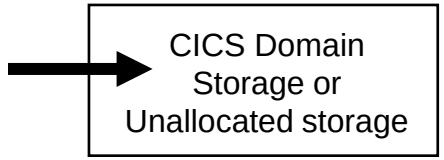
- Allocated task storage will contain an extra 8 bytes on the front and back end
- These areas are known as Storage Check Zones or SCZs
- The are used to assist with storage violation detection

- CICS will only check for storage overlays when the areas are being freed
- If the front and back SCZ do not match the SCA, then a storage violation is detected
- An SVC dump may be produced if set in the CICS dump tables
- If the SIT parameter **STGRCVY = YES** is coded CICS will fix the storage area
- If **STGRCVY = NO** the storage and task are frozen for the life of that CICS run

Storage Account Areas (SAA)

- There is an old format in use for a check zone called an SAA.
- Its use is limited to TIOA areas.
- SAAs are 8 bytes long and are used like SCZ for storage violation detection.
- GETMAIN address is of the leading SAA.
- Point to the current TIOA is in the TCTTE + x’0C’
- Not covered in this presentation

Types of Storage violations

	Task 1	Task 2
Simple Overlay		
Complex		
Undetectable		
Undetectable Complex		

Causes of storage violations?

Some causes of Storage Violations

User programs with incorrect lengths

GETMAIN

TWA / CWA

TCTUA

DFHCOMMAREA (length mismatch)

Linkage Section areas

Incorrect index / subscript value

Runaway index / subscript

No editing of index / subscript

Accessing storage w/o proper addressability

Task / storage availability

Not clearing FREEMAINed pointer

Writing to areas already FREEMAINed

Invalid address as a result of a SET failure

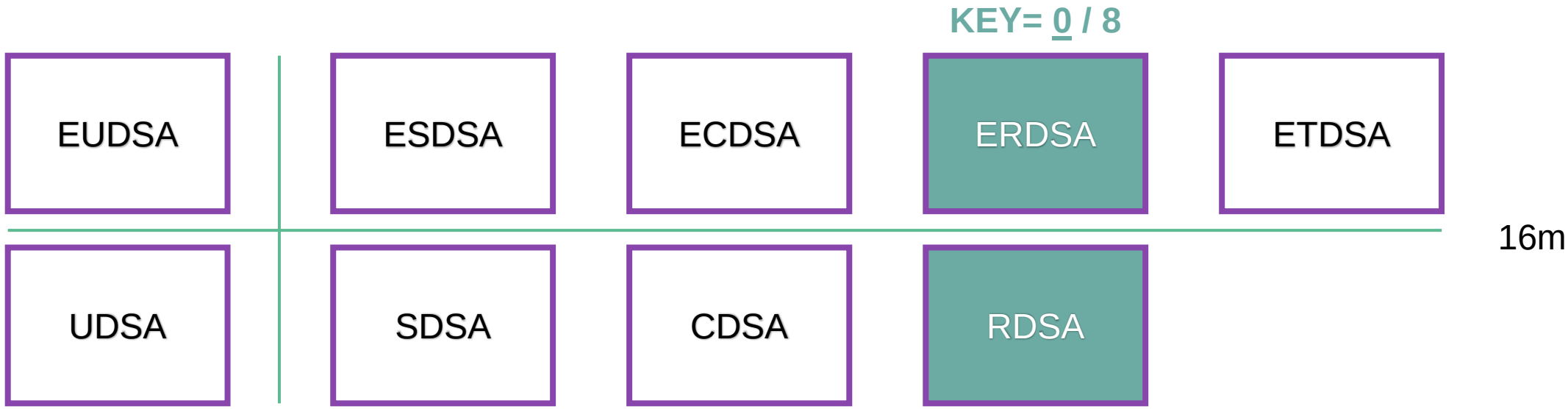
Un-initialized pointers

Maintaining pointers beyond limit

Hand posting ECBs for cancelled or terminated tasks

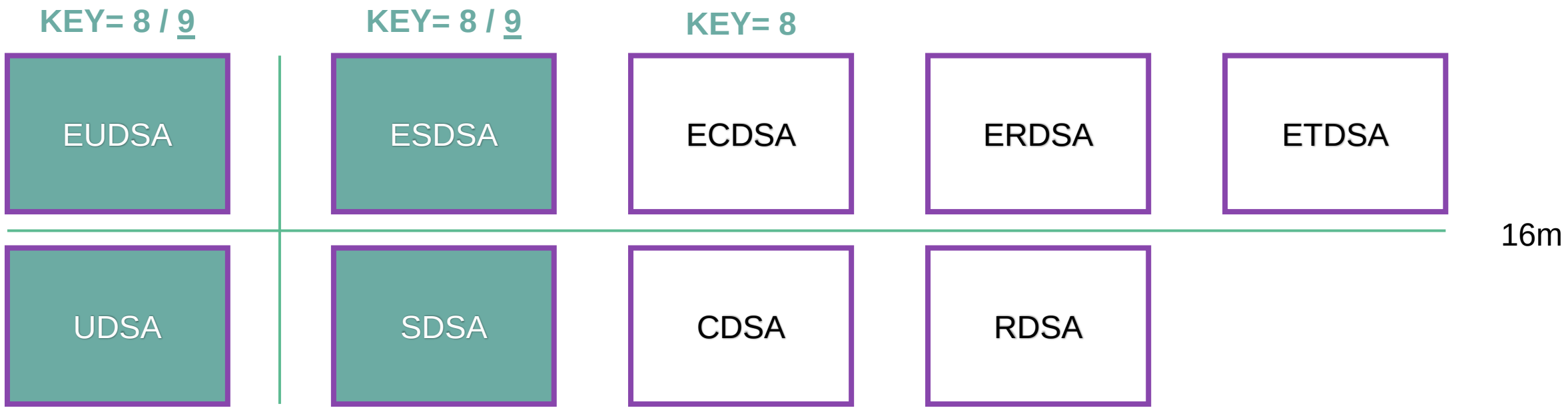
Protecting storage in CICS

Read only program storage



- CICS loads programs that are reentrant in the ERDSA or RDSA based on the DAtalocation parameter in the program definition.
- Non-reentrant programs are loaded in either the ESDSA or the SDSA.
- CICS key programs are loaded in the ECDSA / CDSA when non-reentrant.
- **SIT: RENTPGM = {PROTECT|NOPROTECT}** **KEY = 0 / 8**

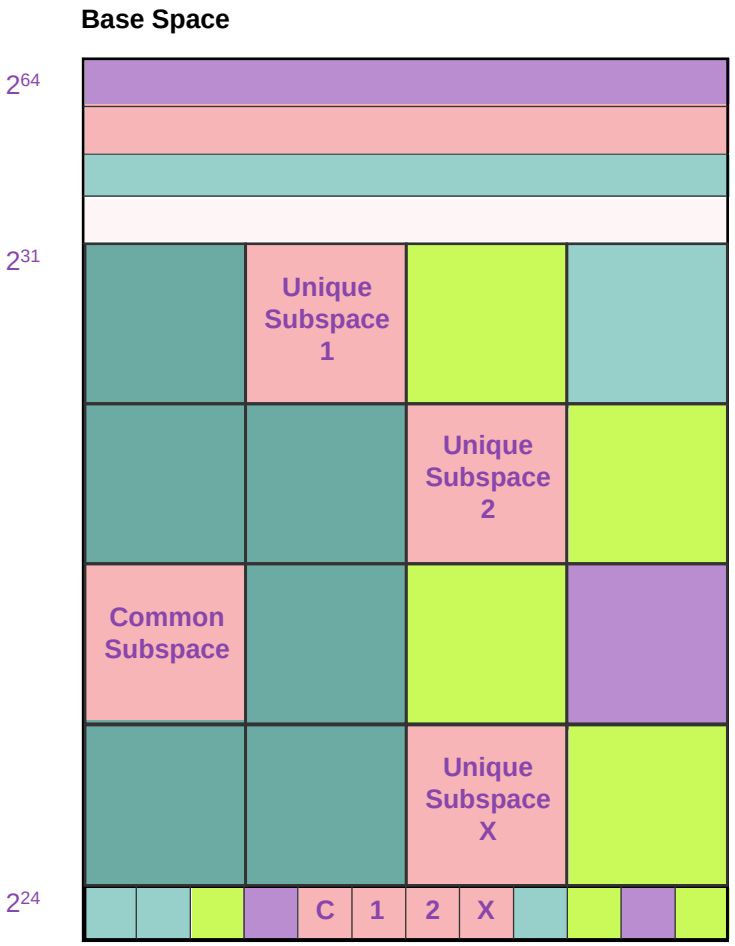
Subsystem storage protection



- Subsystem storage protection separates user storage from CICS key storage.
- Works with the **EXECKey= USER | CICS** parameter on a program definition.
- Prevents a user program from overlying CICS key storage.

- **SIT: STGPROT={NO|YES}** **KEY = 8 / 9**

Transaction Isolation



Only applies to UDSA and EUDSA

Base Space
EXECKEY=CICS

Common Subspace
EXECKEY=USER
ISOLATE=NO

Unique Subspace
EXECKEY=USER
ISOLATE=YES

CICS command protection

SIT: CMDPROT= {YES | NO }

- Used to ensure that CICS was not given an invalid address in a command which could overlay an area that did not belong to the user.
- CICS checks the first byte of an INTO area to make sure the user has access to it.
- CICS checks the address provided for SET commands as well.

Storage protection(s) will only help they do not solve all violations

Storage manager internals

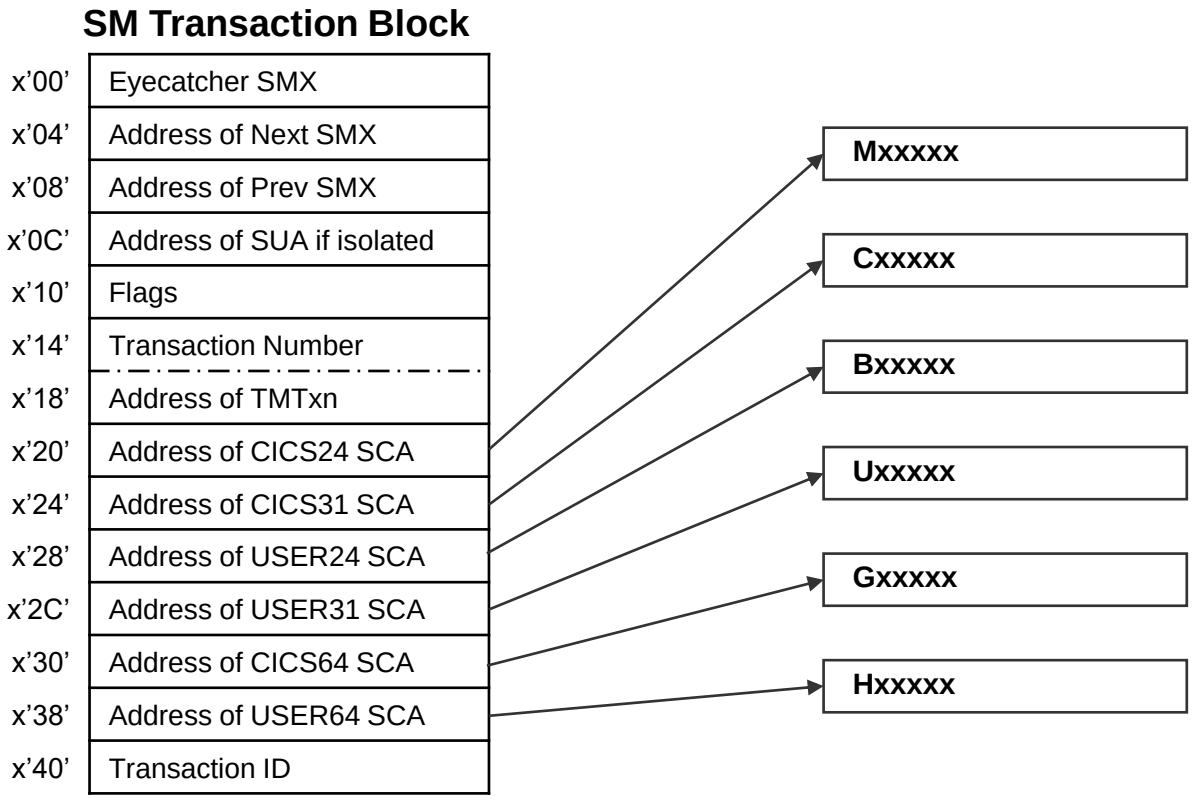
DSA page and extent summary

DSA Summary

	Traniso = Yes		Traniso = No		Cushion
	PAGE SIZE	EXTENT MULTIPLE	PAGE SIZE	EXTENT MULTIPLE	SIZE
CDSA	4k	256k	4k	256k	64k
UDSA	4k	1m	4k	256k	64k
SDSA	4k	256k	4k	256k	64k
RDSA	4k	256k	4k	256k	64k
ECDSA	4k	1m	4k	1m	128k
EUDSA	1m	1m	64k	1m	0k
ESDA	4k	1m	4k	1m	128k
ERDSA	4k	1m	4k	1m	256k
ETSDA	4k	1m	4k	1m	128k
GCDSA	1m	1g	1m	1g	128m
GUDSA	1m	1g	1m	1g	0m
GSDSA	1m	1g	1m	1g	128m

- Extent and page sizes change depending on the TRANISO SIT parameter.
- EUDSA uses the storage short and critical of 256k and 128k for the storage cushion.
- UDSA extent minimum becomes 1m instead of 256k, but page remains 4k.
- EUDSA extent is always a minimum of 1m, but page goes from 64k to 1m.

Storage Manager transaction control block



- The Storage Manager Transaction (SMX) is the anchor for all task related subpools.
- It is chained off the SM Anchor and the Transaction Manager XMTxn control block.
- It is summarized by task in the IPCS VERBX 'SM=1' and addressed in 'XM=1' summaries.

Task variable length SCAs, SCEs and SCFs

SCA

x'00'	Name
x'08'	@Next SCA
x'0C'	@Prev SCA
x'10'	Flags: x'80' Quickcell
x'11'	DSA Access
x'12'	DSA Index
x'18'	Quickcell Related Fields
x'60'	@1st SCE
x'68'	@Last SCE
x'78'	@1st SCF
x'80'	@Last SCF
x'94'	@PPA
x'A7'	Usage x'01' Task x'02' Domain
x'A8'	Chaining x'01' Fixed x'02' Variable
x'A9'	Type x'01' Fixed x'02' Variable
x'AA'	Subpool ID
x'C0'	@SMX
x'C4'	@Subspace Area

SCE

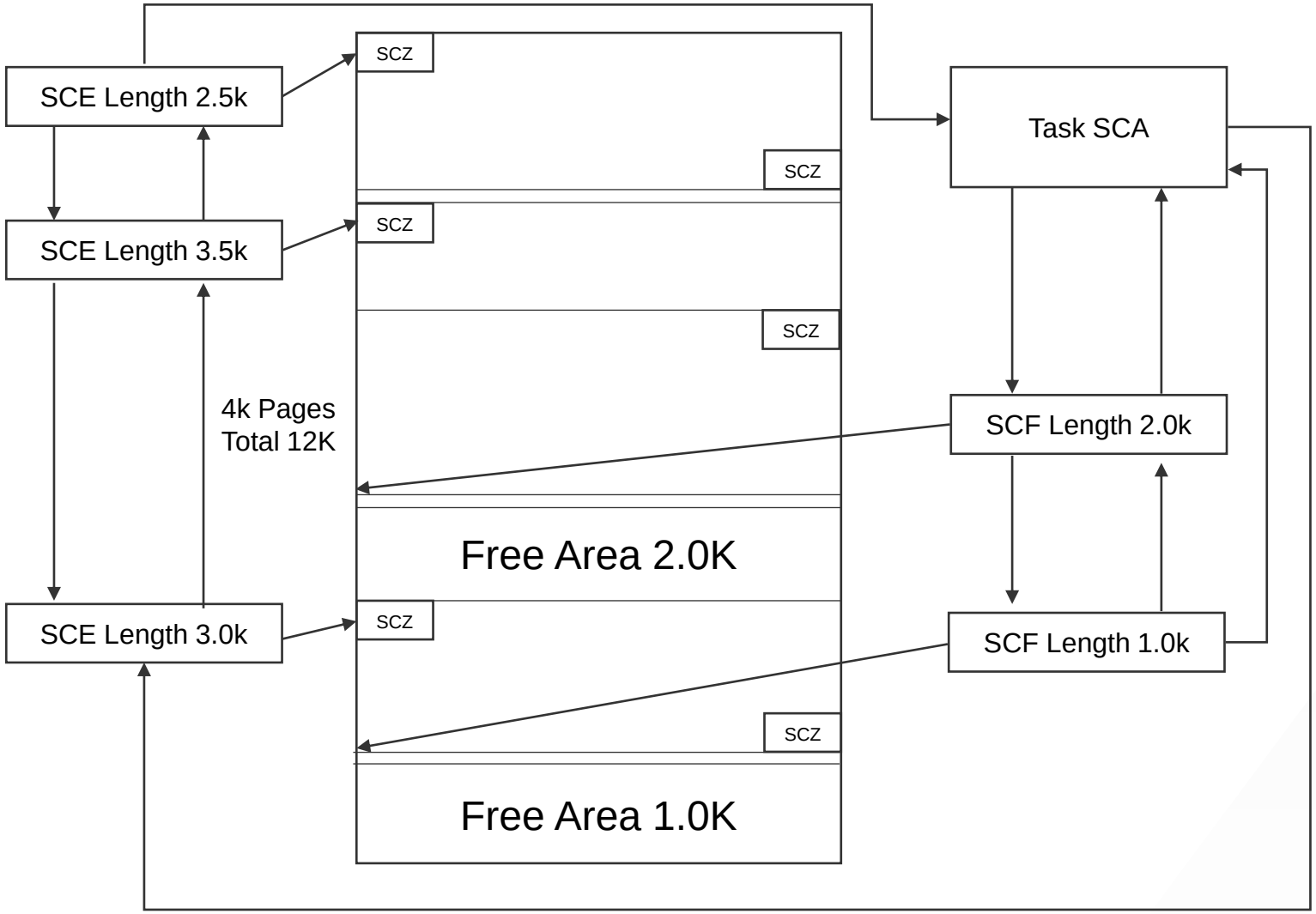
x'00'	@Next SCE
x'08'	@Previous SCA
x'10'	@Allocated Area
x'14'	Length of Allocated Area
x'18'	@PPX
x'20'	Getmain TOD
x'28'	Transaction Number
x'2C'	Tranid

SCF

x'00'	@Next SCF
x'08'	@Previous SCF
x'10'	@Free Area
x'14'	Length of Free Area
x'18'	@PPX
x'1C'	Reserved

- The SCA is an anchor for storage a particular subpool, IPCS VERBX 'SM=3'.
- SCEs represent allocated storage, chained in recently acquired to oldest sequence
- SCFs represent free storage remaining in the page, chained in address sequence.
- In IPCS SCEs and SCFs come right after their associated SCAs.

Task variable length SCAs, SCEs and SCFs



PD12023

example of the SCE/SCF chain for task storage.

Allocated storage: Notes

- The address in an SCE points to the SCZ which matches the subpool name.
- The GETMAIN address points to the first byte of the user area (@SCZ + 8).
- This is important to remember when looking at CICS Trace in available.
- Recall that an SCZ is appended on the front and back end of user storage.
- Therefore, the size of the GETMAIN will be at least 16 bytes longer.
- GETMAINS are rounded on a 16-byte boundary, minimum GETMAIN is 32 bytes.
- There are no chain pointers in SCZs, nothing to break like the old SAAs.
- CICS checks virtual storage on FREEMAIN or task end.
- If front and backend SCZ do not match SCA value a storage violation is detected.
- Task may end normally or may ABEND, if ABEND, look a transaction dump.

Storage violation dump analysis

So where do we start?

- Exception Trace cannot be turned off so we can always start there
- In the SM0102 dump use option 6 and type:
  - **VERBX DFHPDvrm 'TR=3,TRS=<EXCEPTION>' (CICS v5.6 vrm is 730)**
  - **Find *EXC* 15 Last**
- That should take you to the last Exception entry our storage violation
- A simple internet search on the trace point id (SM 0F0C) returns

SM 0F0C	DFHSMAR	Exc	Storage check failure	1	SMAR parameter list
				2	Address of storage element
				3	Length of storage element
				4	First 512-bytes (max) of storage element
				5	Last 512-bytes (max) of storage element
				6	Data preceding storage element (1K max)
				7	Data following storage element (1K max)

- Record the violated address and length from items 2 & 3
- Review items 4 & 5 to find out the SCZ value that was overlaid

Dump Summary

- Every time VERBX is used under option 6 a Dump Summary is the first area displayed in the view.
- In the case of a storage violation, it will have details about the transaction.

```
=== DUMP SUMMARY
   DUMPID:    41/0003
   DUMPCODE:  SM0102
   DATE/TIME: 14/07/21 09:48:54 (LOCAL)
   MESSAGE:   DFHSM0102 FUWFWAR PAY1 00202 A storage violation (code X'0F0C') has been detected by module DFHSMAR.
   SYMPTOMS:  PIDS/5655Y0400 LVLS/730 MS/DFHSM0102 RIDS/DFHSMAR PTFS/HCI7300 PRCS/00000F0C
   TITLE:     (None)
   CALLER:    (None)
   ASID:      X'00B5'
```

- In the message we can see the message code, APPLID, Transaction id, Task number and message.
- In the message itself is the code x’0F0C’ which is our *EXC* message.

Transaction related trace

- If trace is turned on and the trace table is large enough, we can find the original GETMAIN requested for the violated storage within the tasks trace.
  - **VERBX DFHPDvrm ‘TR=1,TRS=<TASKID=xxxxxxx>’ (xxxxxxx is task#)**
  - **FIND the address of the violated storage +8 (Address after SCZ)**

- Note the trace entry number of the GETMAIN response and re-issue:
  - **VERBX DFHPDvrm ‘TR=1’**
  - **FIND the trace entry number (=0923083 in our case)**
- Page forward till you hit the *EXC* SV trace entry noting all the task numbers.
- If no other task ran, or only system tasks then this is a simple overlay.
- Note the programs, LINKs and LOADs one of them is the culprit.
- Also, check if the transaction completed successfully or abended.

Other useful Domains

- Other domains may or may not contain data for the violated task, it will depend on whether the TCA storage has already been freed.
  - Try: **VERBX DFHPDvrm ‘AP=1’ or ‘APS=<TASKID=xxxxxxx>’**
- If the task is not present the storage has already been freed, if this violation occurred at a FREEMAIN, we would see data.
- If present, then another area of interest is Program Manager.
  - Try: **VERBX DFHPDvrm ‘PG=1’ → then FIND ‘PTA Summary’**
- It will give you details of the program link stack as well as commarea addresses and pointers to Channel/Container storage.
- If not, the Transaction Manager will still have some details of the transaction.
  - Try: **VERBX DFHPDvrm ‘XM=1’**
- Using **VERBX DFHPDvrm ‘KE=1’** will show you the running task and its stack storage but will not have the details of the task if the TCA was freed.

Finding the violated storage

- Finding the extent of the violation and reviewing the data in the area can sometimes lead to quicker resolution of the error.
  - Issue: **VERBX DFHPDvrm 'SM=2' (we saw 'SM=1' summary earlier)**
  - Next: **FIND SCA.ZXXXXXXX** where ZXXXXXXX is the SCZ of the violated storage
- Following the SCA will be a list of SCEs and SCFs, go to the last SCE.
- Move upward through the SCEs till you find the address and length of the violated area. The address is at **X'10'** in the SCE and the length is at **X'1C'**.
- Based on whether the front end SCZ or backend SCZ or both are violated, check the storage around the violated area to find the complete violation.
- On the command line issue: **IPCS LIST address LEN(x'length')** to keep your position in the current page but jump to the storage. Using **PF3** will take you out of storage and back to where you left off.
- This way you can easily see the SCZ at the top and bottom of the area.

Storage Violation Trap (CSFE)

- CICS has a built-in storage violation trap know as CSFE.
- Turned on via the SIT: **CHKSTSK=CURRENT** or **CHKSTRM=CURRENT**
- Manually as a transaction via: **CSFE DEBUG, CHKSKxx=CURRENT**
- Checks all SCZs on the transaction storage chain for the currently running task, before any GETMAIN/FREEMAIN activity.
- Requires trace to be active overhead is higher for all running tasks.
- Produces an **SM0103** dump, with a different ***EXC*** trace such as **SM 0932**.
- It turns itself off when a violation is detected, by default, a dump is produced.
- Will catch a violation within a period closer to when it occurred.
- Last resort as the overhead can be quite high.

Summary

- Storage violations are still the most common failure type for CICS.
- Use the storage protection features available in CICS to reduce the detectable and undetectable storage violations.
- When they occur, try and identify the entire violation area to see if the data is recognizable from an application perspective.
- Try and identify all programs that ran from the time of the GETMAIN till the violation, one of them must be the culprit.
- Use the storage violation trap CSFE sparingly to help identify the failure closer to when the overlay occurred.
- When all else fails, IBM is always ready to help!

Rocket®

LEGACY POWERS LEGENDARY

Your feedback is important!