IBM Software Group

Anatomy of a CICS Storage Violation

Paul Albrecht (albrechp@us.ibm.com)
CICS Level II Support - Advisory Software Engineer
24 November 2010

WebSphere® Support Technical Exchange

Agenda

- CICS user storage layouts for a transaction.
  - How is storage laid out?
  - What information are in those areas?
- CICS, Language Environment, COBOL control blocks.
  - Which ones to look at and where to find them.
- COBOL Working Storage Sections.
  - What is a Working Storage Section?
  - What is it used for?
- COBOL Linkage Sections.
  - What is a Linkage Section?
  - What is it used for?
- Diagnostic examples.

Storage Layout for a Transaction

- Let's look at a transaction with 2 link levels, i.e., where program_A linked to program_B.
  - Link Level 1 (program-A)

0005F080 :TCA

0005F494 : SysEIB

00140008 : EIUS
001400D0 : UserEIB

15130000 (Storage area start)

LE Thread Workarea
PCB

LE Control Blocks
EDB, ENSM,OCB, MEML
CAA, SMCB, STKH, etc.
COBOL Control Blocks
THC, THDCOM,
CLLE, TGT, etc.
15133AE0 (Storage area end)

151333AE0 (Storage area start)

LE Control Block
HEAP

COBOL Control Block
Working Storage

1513C0E0 (Storage area end)

Link Level 2 (program_B)

1513C0E0 (Storage area start)

LE Control Blocks
EDB, ENSM,OCB, MEML
CAA, SMCB, STKH, etc.

COBOL Control Blocks
THC, THDCOM,
CLLE, TGT, etc.

15144710 (Storage area end)

15144710 (Storage area start)

LE Control Block
HEAP

COBOL Control Block
Working Storage

1514EC80 (Storage area end)

Storage Layout for a Transaction continued

- OK, we know that Working Storage Sections are vulnerable to overlays and if part of Working Storage is passed as a parameter to another COBOL program that vulnerability still exists as we'll see later.

- Using the addresses, you can see what areas are vulnerable.

15130000 LE Thread Workarea PCB MEMLs LE Control Blocks EDB, ENSM,OCB, MEML CAA, SMCB, STKH, etc. COBOL Control Blocks THC, THDCOM, CLLE, TGT, etc. 15133AE0	151333AE0 LE Control Block HEAP **COBOL Control Block** **Working Storage** 1513C0E0	1513C0E0 LE Control Blocks EDB, ENSM,OCB, MEML CAA, SMCB, STKH, etc. COBOL Control Blocks THC, THDCOM, CLLE, TGT, etc. 15144710	15144710 LE Control Block HEAP COBOL Control Block Working Storage 1514EC80

CICS Control Blocks of Interest

- Once we have found the running task number either in the CICS Trace or Kernel you will have the TCA address and the transaction number.
- From there we can find the EIUS, EIBs, EIS, and general storage layouts in the dump with 'AP=3'.

- We can also use the transaction number to find **what programs are active** for the specific transaction along with how many link levels are involved, and their **commarea addresses** with the PTA Summary report that is generated with the 'PG=3' keyword.

```
==PG: PTA SUMMARY FOR TRAN NUM : 00055,  PTA ADDRESS : 14DAC810
 LOG-LVL : 2        SYS-LVL : 0        TASK-LLE : 00000000  PLCB : 14F932C8
 =PG: TASK PLCB SUMMARY
 PROG COBLINK2 LVL   2 PLCB 14F932C8 LD 16714F00 ENT 96714F20 LEN 005CD0 PPTE 15094030 ENV EXEC INV COBLINK1 EXIT
   COMMAREA 1513A1F0 LEN 111C
  PROGRAM: COBLINK2 CPE: 153C0ED0 LIB: DFHRPL    CONCAT: 40
 PROG COBLINK1 LVL   1 PLCB 14F91060 LD 16710000 ENT 96710020 LEN 004EF8 PPTE 153BFF08 ENV EXEC INV CICS     EXIT
  PROGRAM: COBLINK1 CPE: 153C0E00 LIB: DFHRPL    CONCAT: 02
```

Language Environment Control Blocks of Interest

- These control blocks are the usual ‘anchors’ in LE that are formatted out with the LEDATA verbexit.
  - They are already initialized when the application gains control and are useful in determining how much damage LE has acquired after the storage overlay.
    - CAA Common Anchor Area.
    - EDB Enclave Data Block.
    - PCB Process Control Block.
    - ENSM Enclave-level Storage Management block.
      - HANC Heap ANChor Where COBOL’s Working Storage resides.
    - SMCB Storage Management Control Block.
      - STKH STacK Header Where the LE and application DSAs are stored.
    - There are many other LE blocks that are initialized when they are needed.
      - It’s difficult to determine their state, i.e., are they damaged or not initialized yet.
    - DSA Dynamic Storage Area.
      - Can offer additional information when using LE’s formatted traceback output with the DSA( ) paramter.
      - Unless a DSA (address) parameter is used, the DSA traceback will not be formatted out in CICS.

Language Environment Control Blocks of Interest continued

- Once you have found the CAA address, the LE control blocks can be formatted out with VERBX LEDATA 'CAA(xxxxxxxx),ALL'

```
CAA(15140458),ALL
*******************************************************************************
                    LANGUAGE ENVIRONMENT DATA
*******************************************************************************
Language Environment Product 04 V01 R0C.00
TCB: 00000000             LE Level: 18              ASID: 0049
Active Members: COBOL

CEECAA:  15140458
  +000000  FLAG0:00   LANGP:08   BOS:151410F0     EOS:00000000
  +000044  TORC:00000000    TOVF:8064C5B8    ATTN:1513CAE8
  +00015C  HLLEXIT:00000000       HOOK:50C0D064 0DC058C0 C0060DCC
 ...
  +0005B0  BKWD_CHAIN:15140458    TCB@:00000000    SS_TOP_D:7FFFFFFF
  +000808  SS_DSA_U:00000000      DLLFFLAG:00

  CEEPCB:  15132DD0
  +000000  PCBEYE:CEEPCB    SYSTM:03    HRDWR:03    SBSYS:05    FLAG2:98
  +00000C  DBGEH:00000000    DMEMBR:15133008   ZLOD:956F2D00
 ...
  +0001F0  LANGREUSE:00000000 00000000 00000000 00000000 0000
  +000202  REUSEMEMS:00000000 00000000 00000000 00000000 0000

  CEEEDB:  1513E248
  +000000  EYE:CEEEDB       FLAG1:C0    BIPM:00     BPM:00
  +00000B  CREATOR_ID:02    MEMBR:1513F4A0    OPTCB:1513E950
  ...
```

COBOL Control Blocks of Interest

- These control blocks are the usual ‘anchors’ in COBOL.
  - They are already initialized when the application gains control and are useful in determining how much damage COBOL has acquired after the storage overlay.
    - RUNCOM COBOL rununit communication control block.
    - THC Thread Control block (LE ownership).
    - THDCOM COBOL Thread communication area.
    - COBCOM COBOL partition communication control block.
    - CLLE COBOL Load List Entry.
    - TGT Task Global Table.

COBOL Control Blocks of Interest continued

- Once we have the LE control blocks formatted out with VERBX LEDATA exit, the bottom of the report will contain the COBOL Control Blocks.

```
********************************************************************************
                        COBOL ENVIRONMENT DATA
********************************************************************************
     RUNCOM:  15142168
   +000000  IDENT:C3RUNCOM    LENGTH:000002D8   FLAGS:20C60000
   +000010  RU_ID:1513E248    INVK_RSA:15141188
   +000024  MAIN_PGM_ADDR:16714F20       MAIN_PGM_CLLE:15142440
  ...
   +000098  COBOL_ACTIVE:00000000   DDNAME_SORT_CONTROL:.....
   +0001C8  MAIN_ID:COBLINK2
   +000204  ------>:
   +000240  ------>:

     THC:  151420F0
   +000000  IDENT:IGZTHC     FLAGS:00000000    THCHSPL1:1514210C
   +000010  THCHSPL2:15142110       THCHSPL3:15142114
  ...
   +000030  INSP_WRK:00000000       OPEN_FCBS:00000000

     CLLE:  15142440
   +000000  PGMNAME:COBLINK2  TGT_FLAGS:01      LANG_LST:00000000
   +000010  INFO_FLAGS:9481   LOAD_ADDR:96714F20      TGT_ADDR:15142520
  ...
   +000030  THD_STAT:00000001       THD_CNT:00000000
   +000038  OPEN_NON_EXT_FILES:00000000

     TGT:  15142520
   +000048  IDENT:3TGT  LVL:06      FLAGS:68038260    RUNCOM:15142168
   +00005C  COBVEC:0006C09C  #FCBS:00000000    WS_LEN:0000A490
   +000070  SMG_WRK:00000000       CAA:15140458      LEN:000001D0
   +00008C  EXT_FCBS:00000000      OUTDD:SYSOUT      ABINF:1671518E
   +0000FC  TESTINF:151426A8       PGMADDR:16714F20       1STFCB:00000000
   +000114  WS_ADDR:151447D0       1STEXTFCB:00000000

Exiting COBOL Environment Data
Exiting Language Environment Data
```

COBOL Working Storage

- The Working Storage Section.
  - Contains the variables and data that is used by the program / rununit.
  - It's statically allocated within a HEAP segment and exists for the life program / rununit within the transaction.
    - Note the hierarchy of the declarations, i.e., 01, 05, 10, 15
      - These indicate sub-sections of the higher declaration, i.e., the EMPLOYEE table is part of the DEPT-RECORD.

```
WORKING-STORAGE SECTION.
    77  PGM-NAME       PIC X(8).
    01  FILLER         PIC X(23) VALUE 'BEGIN COMPUTATION DATA'.
    01  PGMNM          PIC X(8)  VALUE 'COBDSUB0'.
    01  FILLER         PIC X(24) VALUE 'END OF COMPUTATION DATA'.
    01  FILLER         PIC X(24) VALUE 'START OF DEPT RECORDS  '.
    01  DEPT-RECORD.
      05   EMPLOYEE-TABLE OCCURS 15 TIMES
          ASCENDING KEY IS WAGE-RATE EMPLOYEE-NO
          INDEXED BY A, B.
        10   EMPLOYEE-NAME   PIC X(20).
        10   EMPLOYEE-NO     PIC 9(6).
        10   WAGE-RATE       PIC 9999V99.
        10   WEEK-RECORD OCCURS 52 TIMES
             ASCENDING KEY IS WEEK-NO INDEXED BY C.
          15  WEEK-NO        PIC 99.
          15  AUTH-ABS       PIC 9.
          15  UNAUTH-ABS     PIC 9.
          15  LATE-ARR       PIC 9.
```

COBOL Linkage Section

- The Linkage Section.
  - MAPS the variables and data that is PASSED by another program / rununit.
    - It describes the data that was passed by another program by a COMMAREA.
    - It can describe other areas that it has access to like TIOAs, TWAs, TCTUAs, getmained areas, etc.
      - Again, note the declaration hierarchy, this entire area describes the COMMAREA that will be passed to it.

```
LINKAGE SECTION.
    01  DFHCOMMAREA.
      05   EMPLOYEE-TABLE OCCURS 15 TIMES
           ASCENDING KEY IS WAGE-RATE EMPLOYEE-NO
           INDEXED BY A, B.
        10   EMPLOYEE-NAME   PIC X(20).
        10   EMPLOYEE-NO     PIC 9(6).
        10   WAGE-RATE       PIC 9999V99.
        10   WEEK-RECORD OCCURS 52 TIMES
             ASCENDING KEY IS WEEK-NO INDEXED BY C.
          15  WEEK-NO        PIC 99.
          15  AUTH-ABS       PIC 9.
          15  UNAUTH-ABS     PIC 9.
          15  LATE-ARR       PIC 9.
      05   EMPLOYEE-HIST   PIC X(4000).
      05   EMPLOYEE-PRO    PIC X(500).
      05   AWARD-HIST      PIC X(1000).
```

Diagnostics

- There are 2 diagnostic exercises.
  - The first one will make point about program checks and 4000 series abends in Language Environment.
  - The second on will have the following goals.
    - Review a typical storage violation.
      - Where the overlay was discovered.
      - The overlay pattern (if possible).
      - The overlay length (if possible).
    - Collect the addresses of the vulnerable areas that the application uses:
      - Working Storage addresses and lengths.
      - Known Linkage Section addresses and lengths.
    - Then tie it all together to determine where and why the overlay occurred.

Diagnostics – Exercise 1

- A customer sends in a dump reporting an abend0C4 at +x'F's from a module called COBLINK2.
- After reviewing the dump we find the program check is in LE module CEEPLOD2 at +x'DEE'

DFHAP0001 IYNXT An abend (code 0C4/AKEA) has occurred at offset X'FFFFFFFF' in module COBLINK2

Transaction LINK was running as task number 00061

```
TCA    0005F700
CAA    15140458

PSW : 478D2000 9568616E 00060010
Regs (0 - F) : 151418A0 151417F0 1513C298 00000001
               1513E248 00000028 40404040 95685380
               15292658 15141C64 1568637F 1568737E
               15140458 151418A0 9568611A 95685380
```

LoadPoint : 15685380 for CEEPLOD2 +x'DEE'

We can perform an additional diagnostic step that may give is a major clue as to what happened….

Diagnostics – Exercise 1 Continued

- We can format the dump with the CICS verbexit's 'AP=3' keyword, and look at the bottom of the report.
  - If we find errors reported by the CICS formatter in the AP domain, it is highly probable that a storage overlay occurred, and that is causing the program check in the Language Environment module.
    - All we did was let the CICS formatter do the work…
  - If we find a message similar to this, then we probably have storage damage in the Application Domain, where the user storage areas are formatted out.

```
=== ERROR MESSAGE INDEX

** Pages containing Error messages :

        81, 87

  Total messages:     2    Informational:     0    Error:     2

-- DFHPD0122I END OF DUMP FOR JOB IYNXT
```

Diagnostics – Exercise 1 Continued

- Using the CICS verbexit's 'AP=3' keyword, searching for DFHPD messages…
  - We're able to map out the storage overlay and find:

```
USER31.00061 15133AE0 USER storage above 16MB

    0000  E4F0F0F0 F0F0F6F1 00000000 00000000  00000000 00000000 00000000 00000000  *U0000061........................*    15133AE0
    0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    15133B00
    0040  00000000 00000000 00000000 00000000  15135C48 00000000 15135990 00000000  *...................*............*    15133B20
    0060  C5D4E7C2 00744000 40404040 40404040  40404040 40404040 00000000 00000000  *EMXB.. .                .........*    15133B40
   ...
    6740 -    781F LINES SAME AS ABOVE                                                                                    1513A220
    7820  00000000 00000000 00000000 40404040  40404040 40404040 40404040 40404040  *............                    *    1513B300
    7840  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513B320
    7860 -    85FF LINES SAME AS ABOVE                                                                                    1513B340

** DFHPD0125  Storage violation detected at 15133AE0. Trailing SAA is invalid.


USER31.00061 1513C0E0 USER storage above 16MB

    0000  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513C0E0
    0020 -    019F LINES SAME AS ABOVE                                                                                    1513C100
    01A0  40404040 40404040 40404040 40404040  40404040 40404040 D7D4C3C2 00280001  *                    PMCB....*    1513C280
    01C0  40404040 40404040 40404040 40404040  60404040 40404040 40404040 40404040  *                -              *    1513C2A0
    01E0  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513C2C0
    0200 -    079F LINES SAME AS ABOVE                                                                                    1513C2E0
    07A0  40404040 40404040 00000000 00000000  C5D5D4C1 00000000 00000000 00000000  *        ........ENMA............*    1513C880
    07C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    1513C8A0

** DFHPD0124  Storage violation detected at 1513C0E0. Leading SAA is invalid.
```

  - This is a storage overlay that can be diagnosed from the CICS perspective also…

Diagnostics – Exercise 1 Conclusion

- The point of Exercise 1
  - If we find a program check and we determine that it's in Language Environment's code, check the AP domain for format errors to see if a CICS detectible storage violation has occurred.
    - CICS has tools like the CSFE storage checker and trace to pursue the problem.
  - The same with LE's 4000 series abends.
    - Abends like the 4088, 4092, 4094, etc… can also indicate a storage overlay.
  - The report from the CICS verbexit's 'AP=3' keyword may offer more clues as to what was happening in the system at the time of the error.

Diagnostics – Exercise 2 Introduction

- This exercise will take the same overlay as in Exercise 1 and carry it further.
  - We will use the LE and COBOL control blocks to uncover the areas that are vulnerable to overlays. These will include the:
    - COBOL Working storage areas.
    - COBOL Linkage storage areas.
      - That may include COMMAREAs, TCTUAs, TWAs, Etc…
  - This will give us a better understanding of why the overlay occurred…

Diagnostics – Exercise 2

- We find that a DFHSM0102 A storage violation (code X'030B') has occurred.
  - Trace is not on, but from the exception entry:

```
SM 030B SMGF  *EXC* - Storage_check_failed_on_freemain_request - FUNCTION(FREEMAIN) ADDRESS(1513C0E8) REMARK(LE_RUWA) STORAGE_CLAS
                (TASK31)

              TASK-00061 KE_NUM-005B TCB-QR   /008C0E88 RET-9460ABB6 TIME-13:40:17.9624863833 INTERVAL-00.0000006093        =05727
                1-0000  00480000 0000000E 00000000 00000000  B4180000 00000000 02000100 00000000  *................................
                  0020  00000000 1513C0E8 00000000 00000000  00000000 0000D3C5 6DD9E4E6 C1400700  *.......{Y..............LE_RUWA .
                  0040  00000000 00000000                                                        *........
                2-0000  1513C0E0                                                                *..{\
                3-0000  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *
                  0020  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *
                  0040  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *
                  0060  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *
                4-0000  00000000 00000000 E4F0F0F0 F0F0F6F1                                      *........U0000061
```

- We determine that the User storage at location 1513C0E0 has it's leading storage accounting area overlaid with blanks.
- You also see that the storage represents a LE_RUWA area
  - I.E, the LE RunUnit Work Area.

Diagnostics – Exercise 2 Continued

- Let's look at the AP domain to determine the extent of the overlay, and at the bottom of the AP=3 report we find:

=== ERROR MESSAGE INDEX

**** Pages containing Error messages :**

83, 89

Total messages: 2 Informational: 0 Error: 2

-- DFHPD0122I END OF DUMP FOR JOB IYNXT

- OK, what do we know so far?
  - We know that the leading storage accounting area for the LE_RUWA at 1513C0E0 has been overlaid.
  - We know that there are 2 format errors in the AP formatted dump.

Diagnostics – Exercise 2 Continued

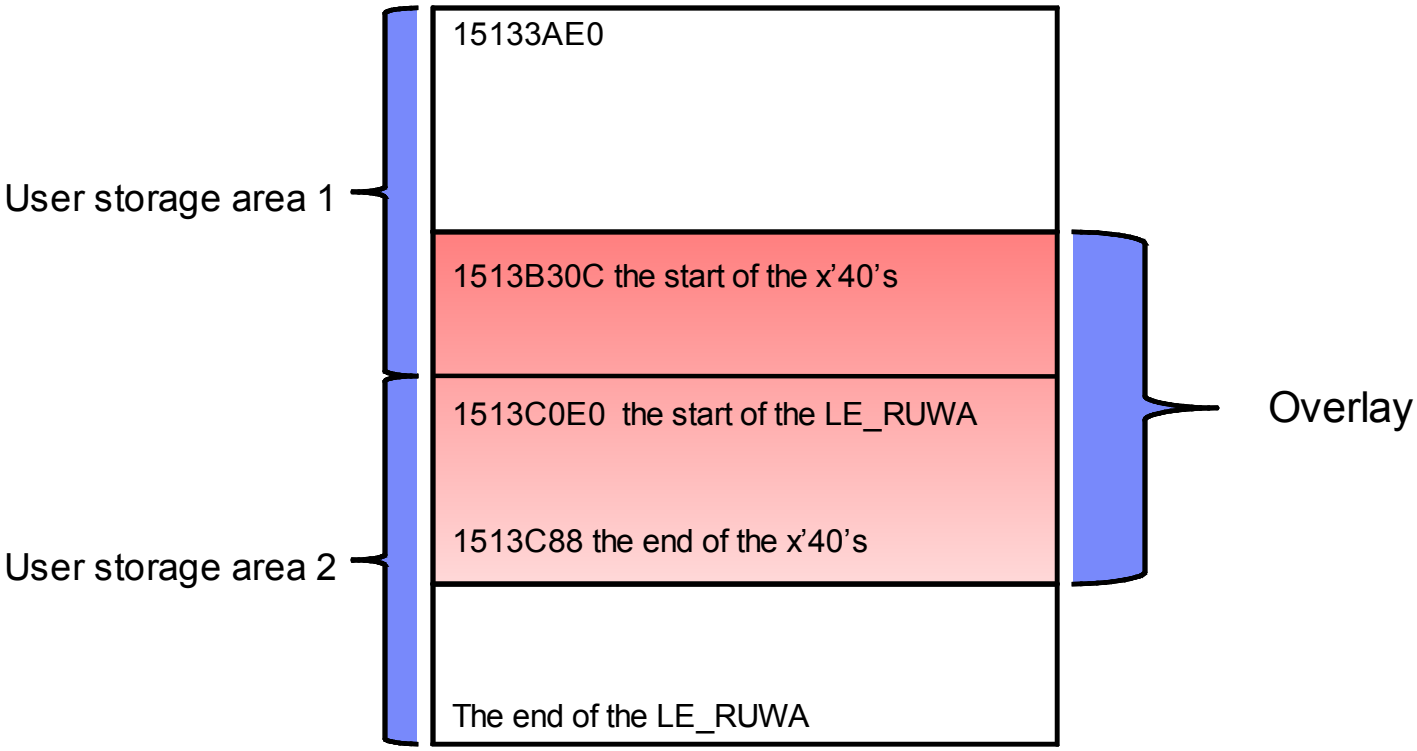
- Looking at the AP domain and the DFHPD error messages you find:

```
USER31.00061 15133AE0 USER storage above 16MB

    0000  E4F0F0F0 F0F0F6F1 00000000 00000000  00000000 00000000 00000000 00000000  *U0000061........................*    15133AE0
    0020  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    15133B00
    0040  00000000 00000000 00000000 00000000  15135C48 00000000 15135990 00000000  *...................*............*    15133B20
    0060  C5D4E7C2 00744000 40404040 40404040  40404040 40404040 00000000 00000000  *EMXB.. .                .......*    15133B40
   ...
    6740 -    781F LINES SAME AS ABOVE                                                                                  1513A220
    7820  00000000 00000000 00000000 40404040  40404040 40404040 40404040 40404040  *............                    *    1513B300
    7840  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513B320
    7860 -    85FF LINES SAME AS ABOVE                                                                                  1513B340

** DFHPD0125  Storage violation detected at 15133AE0. Trailing SAA is invalid.


USER31.00061 1513C0E0 USER storage above 16MB

   0000  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513C0E0
   0020 -    019F LINES SAME AS ABOVE                                                                                  1513C100
   01A0  40404040 40404040 40404040 40404040  40404040 40404040 D7D4C3C2 00280001  *                       PMCB....*    1513C280
   01C0  40404040 40404040 40404040 40404040  60404040 40404040 40404040 40404040  *                 -             *    1513C2A0
   01E0  40404040 40404040 40404040 40404040  40404040 40404040 40404040 40404040  *                               *    1513C2C0
   0200 -    079F LINES SAME AS ABOVE                                                                                  1513C2E0
   07A0  40404040 40404040 00000000 00000000  C5D5D4C1 00000000 00000000 00000000  *        ........ENMA............*    1513C880
   07C0  00000000 00000000 00000000 00000000  00000000 00000000 00000000 00000000  *................................*    1513C8A0

** DFHPD0124  Storage violation detected at 1513C0E0. Leading SAA is invalid.
```

- What do we know so far?
  - We know that the leading storage accounting area for the LE_RUWA at 1513C0E0 has been overlaid with blanks.
  - We know that the trailing storage accounting area for the User storage a 15133AE0 has been overlaid with blanks.
  - It looks like the pattern of blanks extends from 1513B30C to 1513C888.

Diagnostics – Exercise 2 Continued

- Here is another way of looking at what we know :



Diagnostics – Exercise 2 Continued

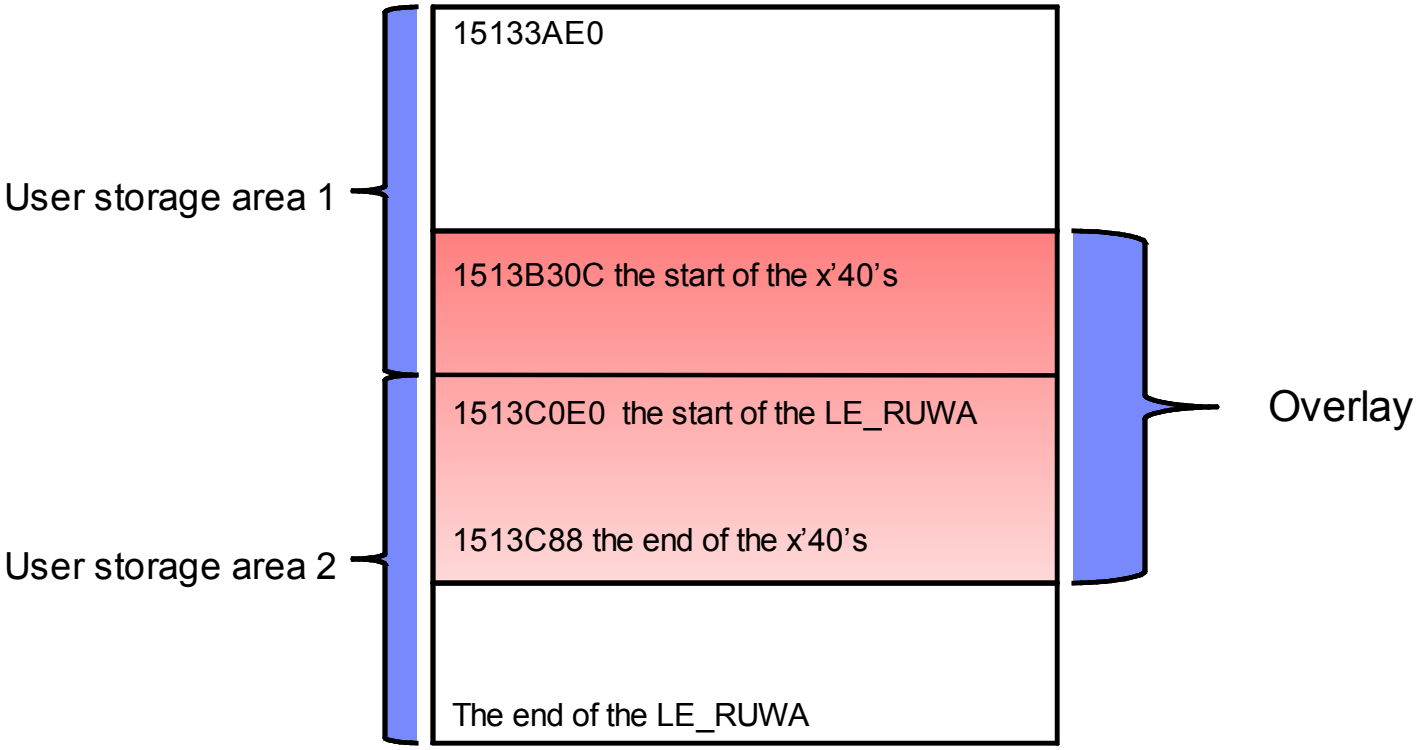
- OK, we know where the overlay is and it's extent. Let's see if we can determine where the overlay starts in relation to the current program's Working Storage area – a vulnerable part of user storage.
- We know that the transaction number is 00061, from there we can access the TCA from the AP or KE formatted domains.
  - The TCA is located at 0005F700, and the transaction is called LINK.
  - The TCA +x'194' contains the LE CAA address 15140458.
- Let's find the Working Storage Area with the - **verbx ledata 'caa(15140458),all'** request.
  - At the bottom of the report we find 1 CLLE for program COBLINK2 and it's associated TGT.

```
 CLLE:  15142440
+000000  PGMNAME:COBLINK2  TGT_FLAGS:01      LANG_LST:00000000
+000010  INFO_FLAGS:9481   LOAD_ADDR:96720DE0      TGT_ADDR:15142520
+00001C  LE_TOKEN:00000000       WSA_ADDR:00000000
+000030  THD_STAT:00000001       THD_CNT:00000000
+000038  OPEN_NON_EXT_FILES:00000000

   TGT:  15142520
+000048  IDENT:3TGT  LVL:06      FLAGS:68038260    RUNCOM:15142168
+00005C  COBVEC:0006E09C   #FCBS:00000000     WS_LEN:0000A490
+000070  SMG_WRK:00000000       CAA:15140458      LEN:000001D0
+00008C  EXT_FCBS:00000000      OUTDD:SYSOUT      ABINF:16721152
+0000FC  TESTINF:151426A8       PGMADDR:16720DE0       1STFCB:00000000
+000114  WS_ADDR:151447D0       1STEXTFCB:00000000
```

- OK, since the Working Storage address of 151447D0 is beyond the overlay boundaries of 1513B30C and 1513C88 we know that it's not involved with the overlay.
- We have just eliminated COBLINK2's Working Storage Area.

Diagnostics – Exercise 2 Continued

- OK, what do we know now.
  - We know where the overlay is and it's extent.
  - We know that the current program's Working Storage Area at 151447D0 is not involved.



Diagnostics – Exercise 2 Continued

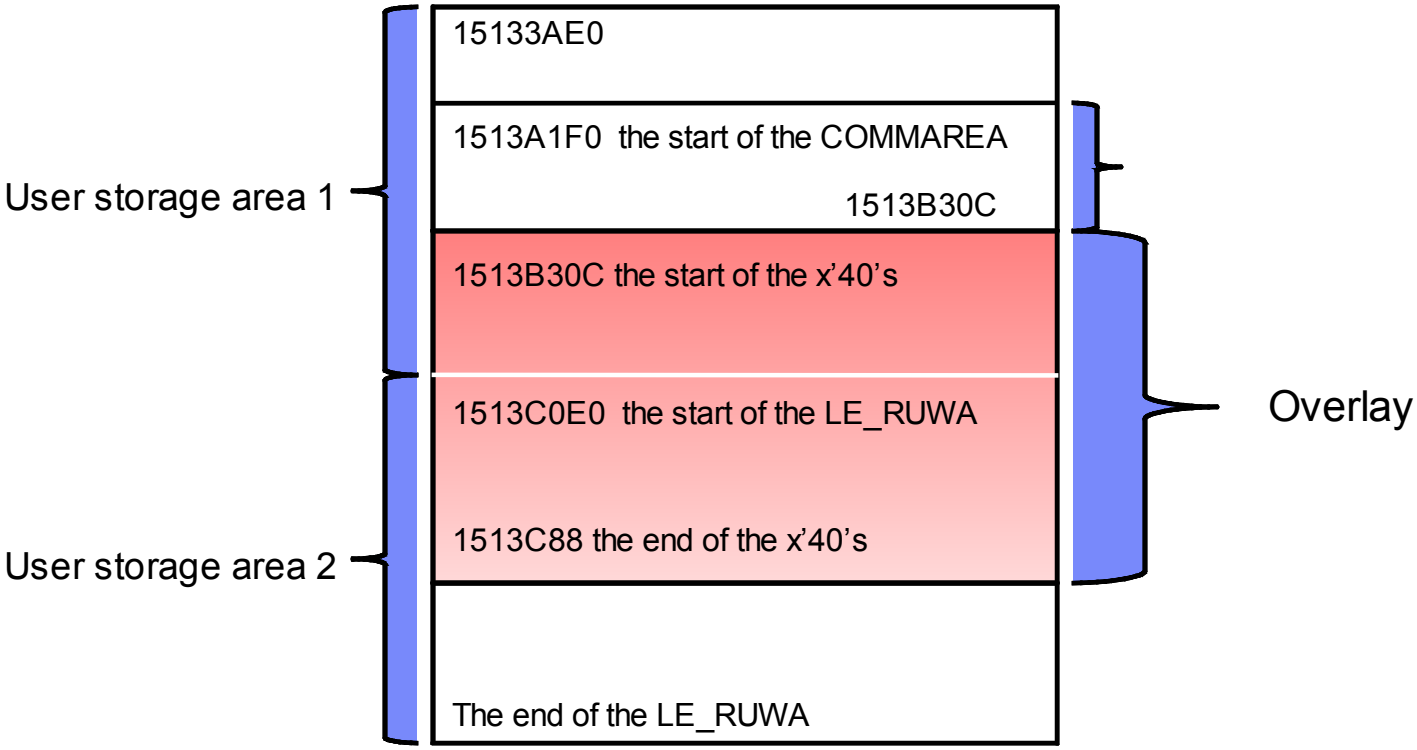
- We still don't know where the overlay originated from…
  - Perhaps there is another program involved.
  - We know that the transaction number is 00061, and maybe it was linked to by another program.
  - Let's check the dump with the PG=3 keyword to see what link levels were active for this transaction.

```
==PG: PTA SUMMARY FOR TRAN NUM : 00061,  PTA ADDRESS : 14DAC810
LOG-LVL : 2          SYS-LVL : 0          TASK-LLE : 1503C2F0  PLCB : 14F932C8

=PG: TASK LLE SUMMARY
LLE-ADDR     PROGRAM      PPTE-ADD
1503C2F0     COBDSUB1     153B7CA8

=PG: TASK PLCB SUMMARY

PROG COBLINK2 LVL   2 PLCB 14F932C8 LD 16720DC0 ENT 96720DE0 LEN 005EF8 PPTE 1504B030 ENV EXEC INV COBLINK1 EXIT
  COMMAREA 1513A1F0 LEN 111C
 PROGRAM: COBLINK2 CPE: 153B8ED0 LIB: DFHRPL   CONCAT: 40

PROG COBLINK1 LVL   1 PLCB 14F91060 LD 16710000 ENT 96710020 LEN 004EF8 PPTE 153B7F08 ENV EXEC INV CICS     EXIT
 PROGRAM: COBLINK1 CPE: 153B8E00 LIB: DFHRPL   CONCAT: 02
```

- OK we know now that COBLINK1 linked to COBLINK2 with a COMMAREA located at 1513A1F0 and a length of x'111C'.
- We also see that the COMMAREA is pointing above the overlaid area.

Diagnostics – Exercise 2 Continued

- OK, what do we know now.
  - We know where the overlay is and it's extent.
  - We know that the current program's Working Storage Area for COBLINK2 at 151447D0 is not involved.
  - We know that COBLINK1 linked to COBLINK2 with a COMMAREA and that COMMAREA is right above the overlay.
    - (The COMMAREA address is 1513A1F0 and it's length is 111C, so 1513A1F0 + 111C = 1513B30C .)



Diagnostics – Exercise 2 Continued

- The COMMAREA looks highly suspicious, yet it ends just before the overlay pattern starts.
  - And then again, it ends just prior to the overlay pattern… So let's be sure…
  - We know that COBLINK1 linked to COBLINK2 and that COBLINK2's Working Storage is not involved.
- Let's look at COBLINK1's Working Storage area.
  - Each CICS link level will have a LE CAA representing the main COBOL program.
    - We can **access that prior CAA** through LE's **EBD_Parent Field**.
    - Format the dump with the LE verbexit using the current CAA from the TCA .
      - This would be COBLINK2's CAA.
    - Then find the EDB.

```
CEEEDB:  1513E248
  +000000  EYE:CEEEDB          FLAG1:C0     BIPM:00      BPM:00
  +00000B  CREATOR_ID:02       MEMBR:1513F4A0     OPTCB:1513E950
  +000014  URC:00000000        RSNCD:00000000     DBGEH:00000000
  +000020  BANHP:1513E758      BBEHP:1513E788     BCELV:956878E8
  +00002C  PCB:15132DD0        ELIST:00000000     PL_ASTRPTR:00000000
  +000038  DEFPLPTR:00140050          CXIT_PAGE:00000000
  +000040  DEBUG_TERMID:40404040   PARENT:15137E58    R13_PARENT:00000000
  +000054  LEOV:95620568       ENVAR:1513C570     ENVIRON:1513E2A0
```

- Now you're ready to format the LE and COBOL control blocks for COBLINK1
  - **verbx ledata 'caa(15137E58),all'**

Diagnostics – Exercise 2 Continued

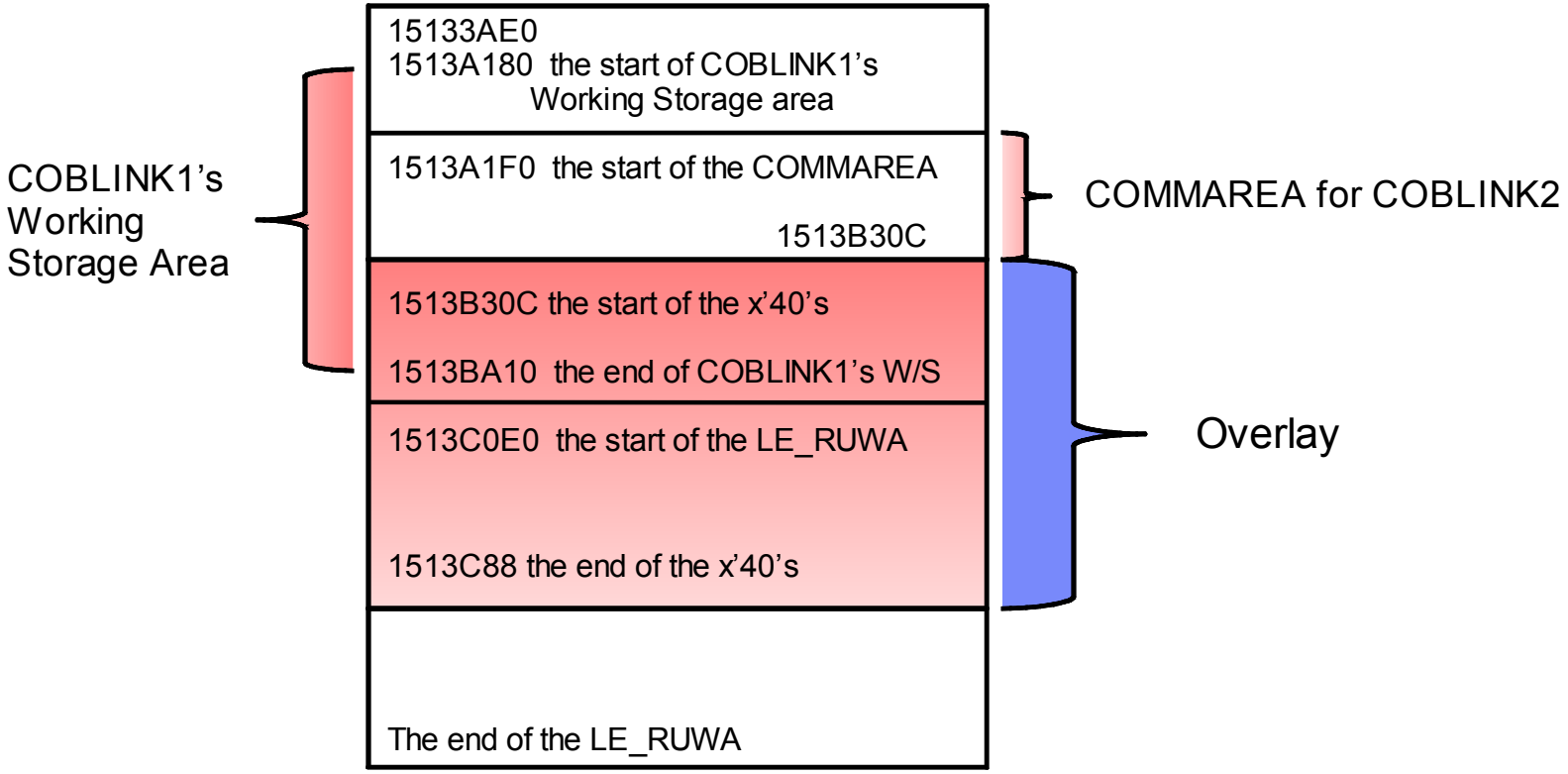
- The verbx ledata with the prior CAA (i.e, COBLINK1's CAA) shows the following at the bottom of the report.

```
 CLLE:  15139E40
+000000  PGMNAME:COBLINK1  TGT_FLAGS:01       LANG_LST:00000000
+000010  INFO_FLAGS:9481   LOAD_ADDR:96710020      TGT_ADDR:15139F20
+00001C  LE_TOKEN:00000000       WSA_ADDR:00000000
+000030  THD_STAT:00000001       THD_CNT:00000000
+000038  OPEN_NON_EXT_FILES:00000000

   TGT:  15139F20
+000048  IDENT:3TGT  LVL:06       FLAGS:68038260    RUNCOM:15139B68
+00005C  COBVEC:0006E09C   #FCBS:00000000    WS_LEN:00001890
+000070  SMG_WRK:00000000        CAA:15137E58      LEN:00000198
+00008C  EXT_FCBS:00000000       OUTDD:SYSOUT      ABINF:1671022E
+0000FC  TESTINF:1513A070        PGMADDR:16710020       1STFCB:00000000
+000114  WS_ADDR:1513A180        1STEXTFCB:00000000
```

- OK we see that COBLINK1's Working Storage begins at 1513A180 and has a length of x'1890'.
- We also see that it starts above the overlay pattern and above the COMMAREA that was passed to COBLINK2.

Diagnostics – Exercise 2 Continued

- OK, what do we know now.
  - We know that the Working Storage Area for COBLINK1 at 1513A180 is involved since the overlay starts at +x'118C' into COBLINK1's Working Storage Area.
  - We know that COBLINK1 linked to COBLINK2 with a COMMAREA and that suspected COMMAREA contained part of COBLINK1's Working Storage, so let's take a look at the COMMAREA as defined by COBLINK2.



Diagnostics – Exercise 2 Continued

- OK, we know that COBLINK1's Working Storage is overlaid and we know that the COMMAREA it passed to COBLINK2 resides just above the overlay pattern.
- What we don't know is whether COBLINK1 overlaid it's own Working Storage or if COBLINK2 has mapped the COMMAREA incorrectly in it's LINKAGE SECTION.
  - We can find out how COBLINK2 is mapping the COMMAREA by using COBLINK2's TGT.
    - The TGT is comprised of 2 sections.
      - A static section that remains constant.
      - And a dynamic section where it's size depends on how big the Working Storage and Linkage Sections are.
      - Here is a TGT map taken from the compile of COBLINK2 for illustrative purposes.

```
*** TGT MEMORY MAP ***
                    TGTLOC

                    000000  RESERVED - 72 BYTES
                    000048  TGT IDENTIFIER
                    00004C  RESERVED - 4 BYTES
                    ...
                    000064  NUMBER OF FCB'S
                    000068  WORKING-STORAGE LENGTH
                    00006C  RESERVED - 4 BYTES
                    ...
                    000110  POINTER TO FIRST FCB CELL
                    000114  WORKING-STORAGE ADDRESS
                    000118  POINTER TO FIRST SECONDARY FCB CELL

                    *** VARIABLE PORTION OF TGT ***

                    000124  BASE LOCATORS FOR SPECIAL REGISTERS
                    00012C  BASE LOCATORS FOR WORKING-STORAGE
                    000158  BASE LOCATORS FOR LINKAGE-SECTION
                    00016C  CLLE ADDR. CELLS FOR CALL LIT. SUB-PGMS.
```

Diagnostics – Exercise 2 Continued

- The Base Locators for Working Storage usually starts at +x'12C' into the TGT.
  - They are addresses used as base registers to access Working Storage.
  - Each address will differ by x'1000' bytes, just like base registers.
  - They are known as BLWs (Base Locators for Working storage).
  - The addresses will provide complete addressability to the Working Storage area.

- The Base Locators for the Linkage Section will start near the end of the BLWs.
  - They are known as BLLs (Base Locators for the Linkage section).
  - They are similar to the BLWs in that they act as base registers to the Linkage Section areas.
  - The BLLs will start just after the BLWs and this depends on how large the Working Storage area is.
  - The BLLs will address the data that was passed to the COBOL application.
    - The EIB and COMMAREA
    - This will be the case for programs that were LINK'd and XCTL'd to.
    - If the programs were statically or dynamically called, they may not contain an EIB or COMMAREA address.
  - The BLL addresses will start with the EIB address and the COMMAREA address (if one was passed).
    - These 2 addresses can be used to find the start of the BLLs.
    - If a COMMAREA was not passed, you still have the EIB address as a reference.
  - The BLLS can also address any other areas that the application needs access to.
    - The TWA, TCTUA, Getmained storage, etc..
    - The address patterns will change in this area depending on how many different areas the application has defined within the Linkage Section.

Diagnostics – Exercise 2 Continued

- OK, that's a lot of information, but we have the TGT address for COBLINK2 from the LEDATA area.
  - Let's take a look at it…

```
TGT : 15142520

15142520.:1514255F. LENGTH(X'40')--All bytes contain X'00'
15142560   00000000   00000000   F3E3C7E3   00000000   | ........3TGT.... |
15142570   06000000   68038260   15142168   0006E09C   | ......b-......\. |
15142580   151426F0   00000000   0000A490   00000000   | ...0......u..... |
15142590   00000000   15144740   00000000   00000000   | ....... ........ |
151425A0   15140458   000001D0   00000000   00000000   | .......}........ |
151425B0   00000000   00000001   E2E8E2D6   E4E34040   | ........SYSOUT   |
151425C0   C9C7E9E2   D9E3C3C4   00000000   00000000   | IGZSRTCD........ |
151425D0.:151425FF. LENGTH(X'30')--All bytes contain X'00'
15142600   00000000   00000000   16720EDC   00000000   | ................ |
15142610   151426DC   15142440   16721152   151426A8   | ....... .......y |
15142620   16720DE0   16720F18   151426D0   16720F08   | ..\.....}....... |
15142630   00000000   151447D0   00000000   00000000   | .......}........ |
15142640   00000000   00000000   15144750   151447D0   | ...........&...} |
15142650   151457D0   151467D0   151477D0   151487D0   | ...}...}...}..g} |
15142660   151497D0   1514A7D0   1514B7D0   1514C7D0   | ..p}..x}...}..G} |
15142670   1514D7D0   1514E7D0   00000000   001400D0   | ..P}..X}.......} |
15142680   1513A1F0   1513B1F0   1513C1F0   00000000   | ..~0...0..A0.... |
15142690.:1514269F. LENGTH(X'10')--All bytes contain X'00'
151426A0   00000000   00000000   00000000   1672180C   | ................ |
```

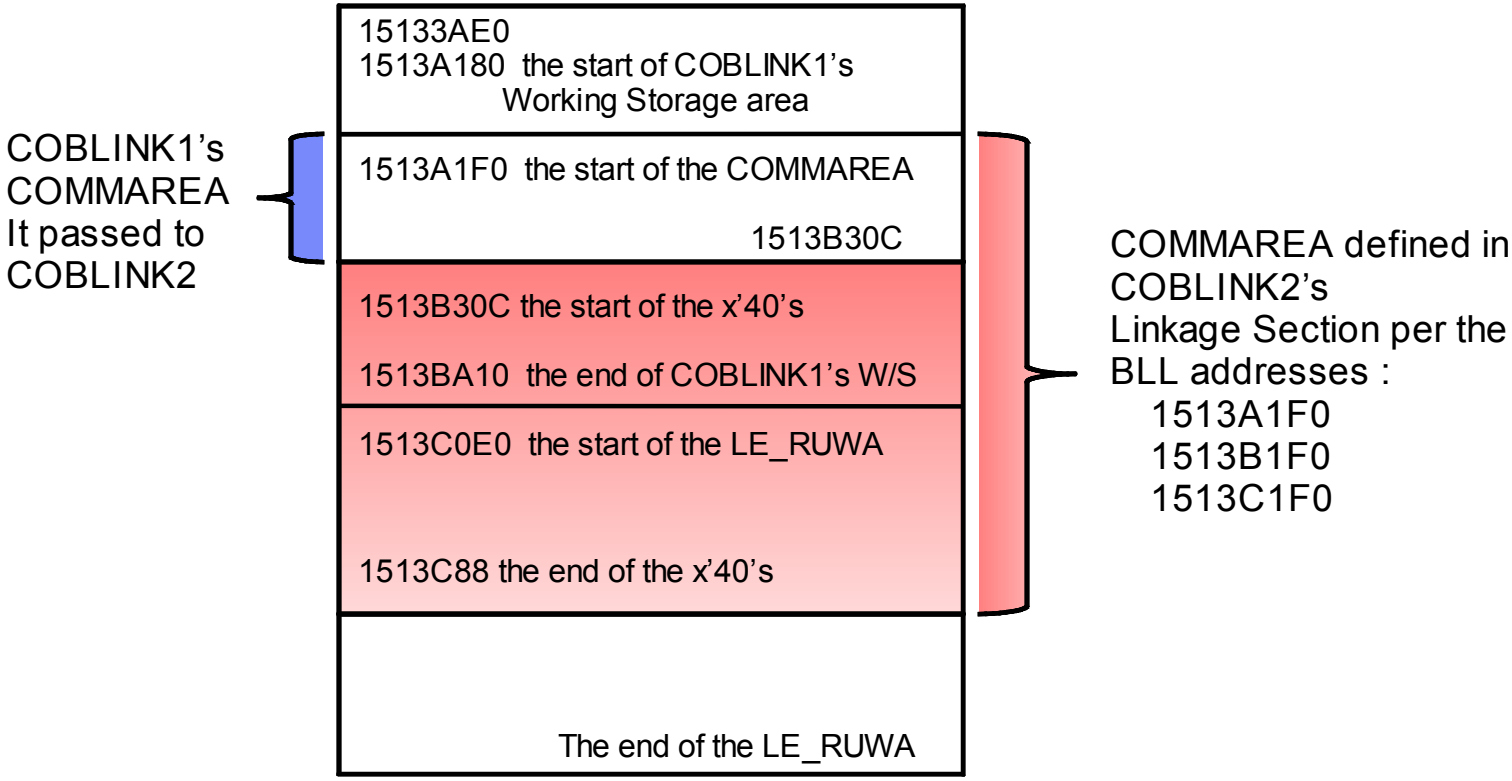
At +x'12C' there's the start of The BLWs for COBLINK2's Working Storage and it looks Like it extends down to 1514E7D0

Just below the BLWs here is the EIB address followed by the COMMAREA address. This is the start of the BLLs for the Linkage Section

OK, here is our COMMAREA address for COBLINK2 followed by 2 other 'base' addresses in x'1000' byte increments. All three addresses are accessing the COMMAREA, **1513A1F0**, **1513B1F0**, and **1513C1F0**.

Diagnostics – Exercise 2 Continued

- OK, what do we know now.
  - We know that the Working Storage Area for COBLINK1 at 1513A180 is involved since the overlay starts at +x'118C' into it and part of the Working Storage Area was passed as a COMMAREA.
  - And we also know that COBLINK2's Linkage Section has defined the COMMAREA much larger than what was passed to it. Since COBLINK2's COMMAREA definition is incorrect, it's a safe bet that COBLINK2 did the damage.



Diagnostics – Exercise 2 Conclusion

- It looks like COBLINK2 has overlaid COBLINK1's Working Storage Area.
  - We have proof that COBLINK2 has defined a COMMAREA much larger that what it was sent, and has used that additional area.
  - The end result was :
    - A program check in LE's loader code from example 1.
    - A SM0102 storage violation from example 2.
- We used the CICS, LE, and COBOL control blocks to look at the overlay in greater detail.
  - The CICS AP domain was used to map the overlay and access the LE CAA.
  - The CICS PG domain was used to show the link levels and commareas involved.
  - The LEDATA was used to.
    - Find the COBOL CLLEs and associated TGTs for the application environment.
    - Find the EDB so that we could find the application environment at other link levels.
- This gave us the pinpoint diagnostics without.
  - Without looking at a trace (usually there isn't one in a production environment).
  - Without relying on the CSFE storage checker (usually can't be run in production).
  - And without looking at the applications.

Additional Information

- Scott McClure's WSTE **Debugging Storage Violations in CICS.**
  - http://www-01.ibm.com/support/docview.wss?uid=swg27007891
- Paul Albrecht's WSTE **Serviceability for COBOL working storage in the CICS environment.**
  - http://www-01.ibm.com/support/docview.wss?uid=swg27017236
- The CICS TS R4.1 Info Center
  - http://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp
- The CICS TS Problem Determination Guide *SC34-6826-00*
- The Language Environment Debugging Guide *GA22-7560-08*

Additional Product Resources

- WebSphere and CICS Support blog
  https://www.ibm.com/developerworks/mydeveloperworks/blogs/aimsupport/?lang=en
- IBM_CICS support news on Twitter
  http://www.ibm.com/support/docview.wss?uid=swg21384915
- Track specific CICS APARs or CICS APARs by component id
  http://www.ibm.com/support/docview.wss?uid=swg21422149
- Sign up to receive technical support e-mails
  http://www.ibm.com/software/support/einfo.html
- CICS Featured documents
  http://www.ibm.com/support/docview.wss?uid=swg27006900
- Webcasts for CICS and OMEGAMON
  http://www.ibm.com/support/docview.wss?uid=swg27007244
- CICS Transaction Server Support Web page
  http://www.ibm.com/support/entry/portal/Overview/Software/Other_Software/CICS_Transaction_Server

We Want to Hear From You!

Tell us about what you want to learn

Suggestions for future topics
Improvements and comments about our webcasts
We want to hear everything you have to say!

Please send your suggestions and comments to:
wsehelp@us.ibm.com

Questions and Answers